Προγραμματισμός Λογιστικών Φύλλων (Excel VBA)

Κώστας Καλπάκας
costask@otenet.gr
Κιν.: 6944261906

ΙΝ.ΕΠ. – Νοέμβριος 2010

4η και 5η Ημέρα Επιμόρφωσης

- Εργασία με *Βιβλία* και *Φύλλα* εργασίας – Δημιουργία Γραφημάτων με VBA
- Αποσφαλμάτωση & Διαχείριση Σφαλμάτων (*Debugging & Error Handling*)
- Σύνδεση με Βάσεις Δεδομένων

- Συνεργασία με άλλες εφαρμογές του Office
- Μελέτη Περίπτωσης
- Αξιολόγηση Προγράμματος & Εκπαιδευομένων

4η Ημέρα Επιμόρφωσης

- Εργασία με *Βιβλία* και *Φύλλα* εργασίας - Δημιουργία Γραφημάτων με VBA
- Αποσφαλμάτωση & Διαχείριση Σφαλμάτων (*Debugging & Error Handling*)
- Σύνδεση με Βάσεις Δεδομένων

Διαχείριση Φύλλων & Βιβλίων Εργασίας

Αναδρομή στα Βασικά Αντικείμενα (Οντότητες) του Excel

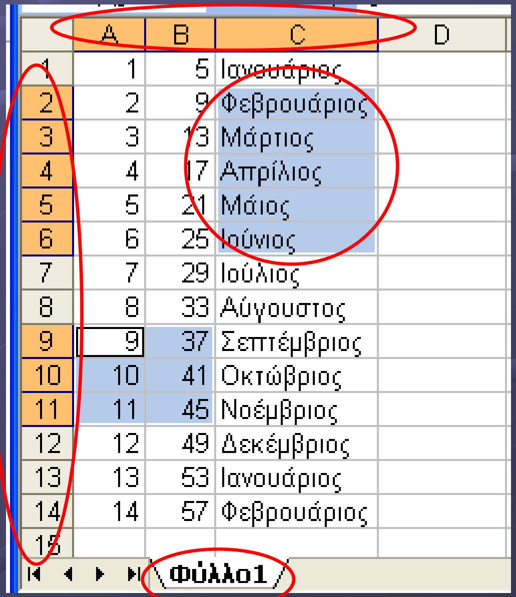
- Ένα αρχείο Excel περιέχει ένα *Βιβλίο Εργασίας* (***Workbook***)

- Ένα Βιβλίο (Workbook) Excel περιέχει ένα ή περισσότερα *Φύλλα Εργασίας* (***WorkSheets***)

- Κάθε Φύλλο Excel αποτελείται από ένα αριθμό *Γραμμών* (*Rows*) και *Στηλών* (*Columns*)

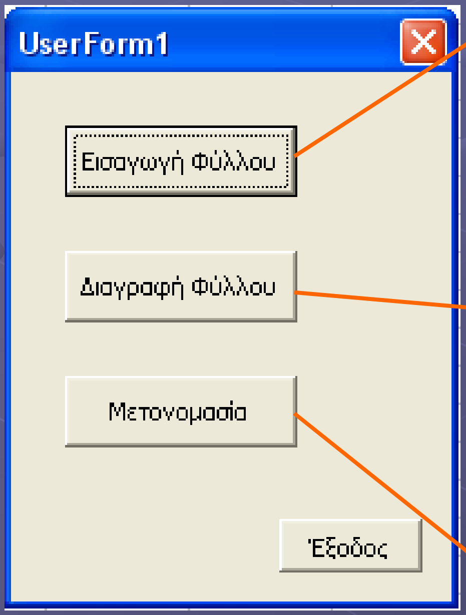
- Το σημείο τομής μιας Γραμμής με μία Στήλη καλείται *Κελί* (*Cell*)

- Ένα ή περισσότερα Κελιά, συνεχόμενα ή μη, συνιστούν μία *Περιοχή* (*Range*)



Διαχείριση Φύλλων & Βιβλίων Εργασίας

Όπως διαχειριζόμαστε όλα τα Αντικείμενα του Excel μέσω της VBA, έτσι μπορούμε να διαχειριστούμε και τα αντικείμενα ***WorkBook*** και ***WorkSheet***, όπως φαίνεται στο παράδειγμα που ακολουθεί:



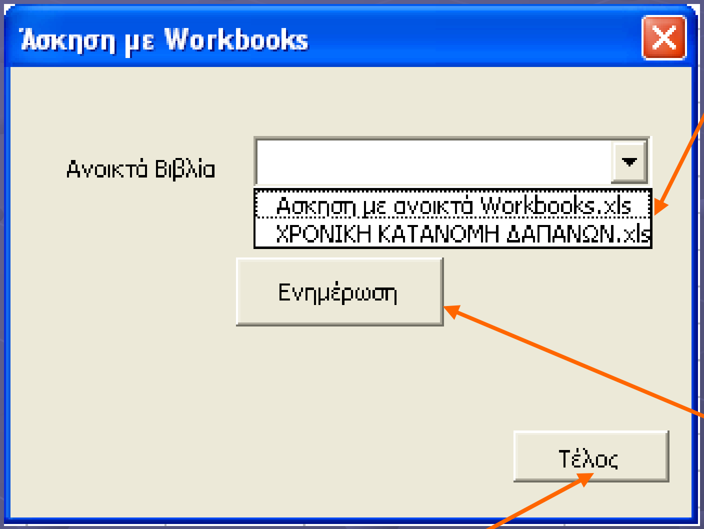

```
Private Sub InsertSheetButton_Click()
   ThisWorkbook.Sheets.Add
   MsgBox “Υπάρχουν : " &  ThisWorkbook.Sheets.Count & " Φύλλα"
End Sub
```

```
Private Sub DeleteSheetButton_Click()
   ThisWorkbook.Sheets(ThisWorkbook.Sheets.Count - 1).Delete
   MsgBox “Απέμειναν : " &  ThisWorkbook.Sheets.Count & " Φύλλα"
End Sub
```

```
Private Sub RenameSheetButton_Click()
   NewName = InputBox(“Δώστε το ΝΈΟ όνομα: ", “Μετονομασία")
   ThisWorkbook.Sheets(1).Name = NewName
End Sub
```

Διαχείριση Φύλλων & Βιβλίων Εργασίας

Στο επόμενο παράδειγμα, γίνεται εντοπισμός όλων των Ανοικτών Βιβλίων Εργασίας και τα ονόματά τους γεμίζουν μια Αναδιπλούμενη Λίστα (dropdown list). Το Βιβλίο που επιλέγεται από την Λίστα γίνεται το Ενεργό Βιβλίο.




```
Private Sub OpenWorkbooksCombo_Change()
    Workbooks(OpenWorkbooksCombo.Value).Activate
End Sub
```

```
Private Sub UpdateButton_Click()
    Dim wb As Workbook
            ' Καθάρισμα της Λίστας
    OpenWorkbooksCombo.Clear
            ' Για κάθε Ανοικτό Workbook
    For Each wb In Workbooks
            ' Προσθέτουμε το όνομά του στη Λίστα
        OpenWorkbooksCombo.AddItem    wb.Name
    Next wb
End Sub
```

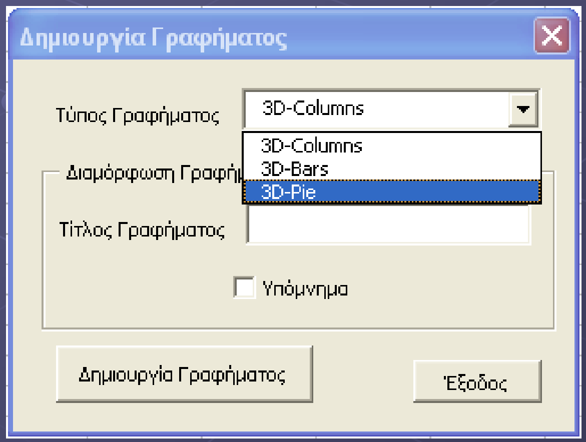
```
Private Sub ExitButton_Click()
        Unload Me
End Sub
```

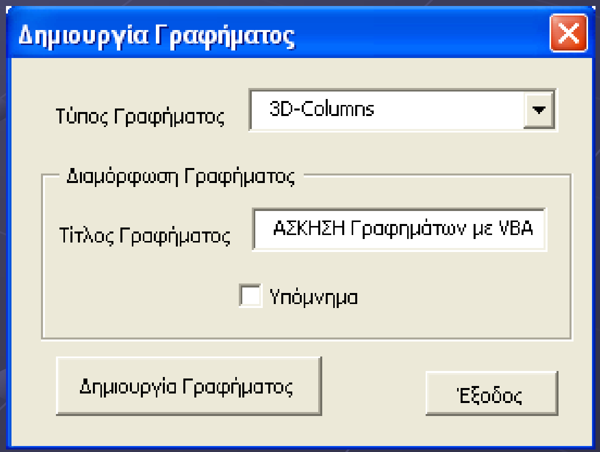
Δημιουργία Γραφημάτων (Charts) με VBA (1/4)

Το Γράφημα (Chart) είναι ένα ακόμα αντικείμενο του Excel που μπορούμε να διαχειριστούμε με την VBA, όπως θα δούμε με την βοήθεια του παρακάτω Παραδείγματος.

Μέσω της Φόρμας, θα γίνεται η επιλογή του «Τύπου Γραφήματος», θα καταχωρείται ο Τίτλος του και θα επιλέγεται αν θα υπάρχει «Υπόμνημα» ή όχι. Κατόπιν, πατώντας το κουμπί «Δημιουργία Γραφήματος» θα δημιουργείται το κατάλληλο γράφημα (με βάση τις επιλογές της Φόρμας), χρησιμοποιώντας τα δεδομένα που περιλαμβάνονται στην επιλεγμένη περιοχή του Ενεργού Φύλλου (ActiveSheet).

	A	B
1	Μήνας	Ποσό
2	Ιανουάριος	1.200
3	Φεβρουάριος	1.400
4	Μάρτιος	1.600
5	Απρίλιος	1.800
6	Μάιος	2.000
7	Ιούνιος	2.200
8	Ιούλιος	1.900
9	Αύγουστος	1.600
10	Σεπτέμβριος	1.300
11	Οκτώβριος	1.000
12	Νοέμβριος	700
13	Δεκέμβριος	400

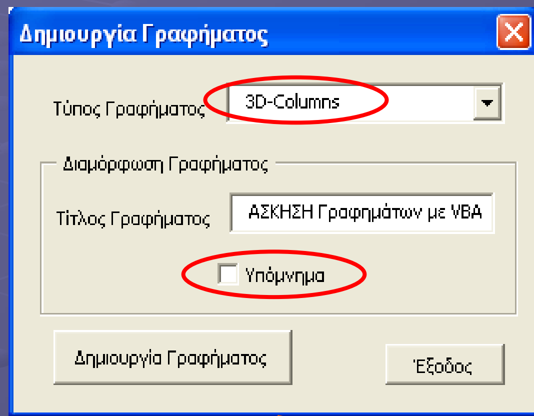


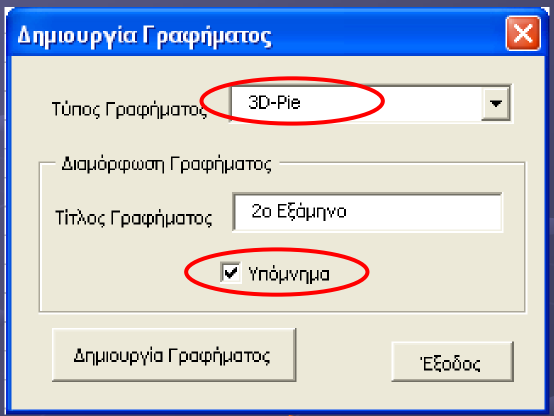


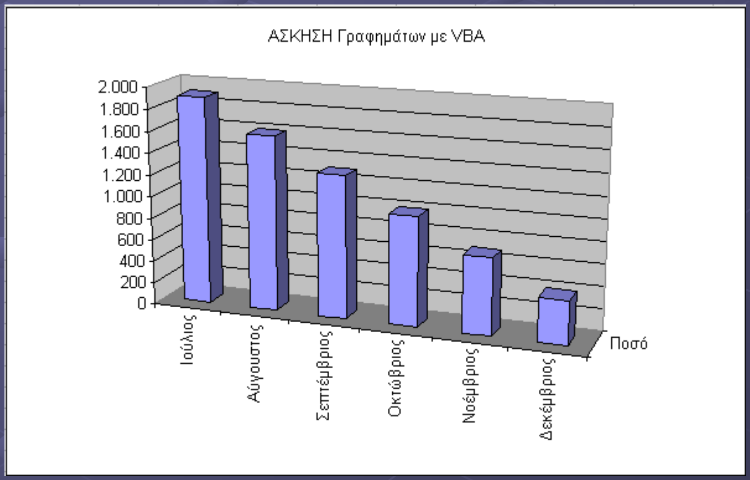
Δημιουργία Γραφημάτων (Charts) με VBA (2/4)

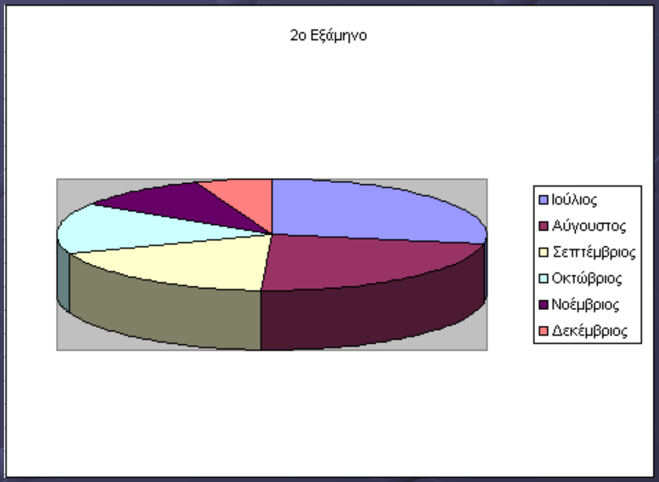
Επιλεγμένη Περιοχή (Range).

	A	B
1	Μήνας	Ποσό
2	Ιανουάριος	1.200
3	Φεβρουάριος	1.400
4	Μάρτιος	1.600
5	Απρίλιος	1.800
6	Μάιος	2.000
7	Ιούνιος	2.200
8	Ιούλιος	1.900
9	Αύγουστος	1.600
10	Σεπτέμβριος	1.300
11	Οκτώβριος	1.000
12	Νοέμβριος	700
13	Δεκέμβριος	400

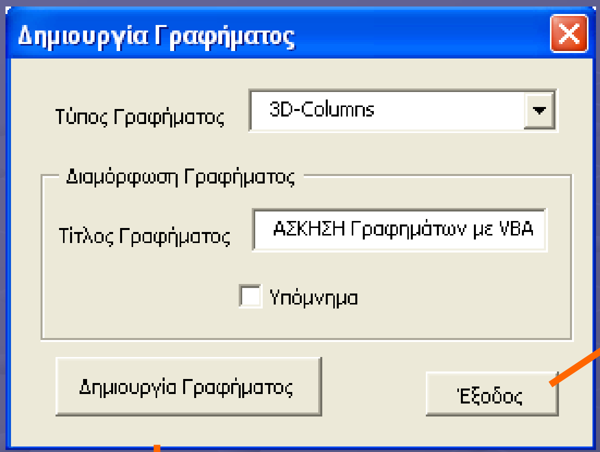








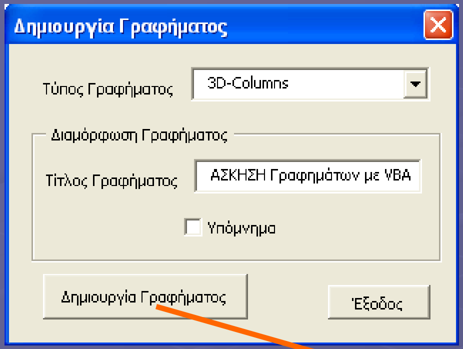
Δημιουργία Γραφημάτων (Charts) με VBA (3/4)




```
' Κλείσιμο της Φόρμας
Private Sub ExitButton_Click()
    Unload Me
End Sub
```

```
' Εκτελείται κάθε φορά που Ενεργοποιείται η Φόρμα και κάνει τον καθορισμό
' των Ιδιοτήτων και την Αρχικοποίηση των στοιχείων της Φόρμας.
Private Sub UserForm_Activate()
    CreateChartForm.Show vbModeless        'Η Φόρμα να μην είναι Modal
    ChartTypeComboBox.Style = fmStyleDropDownList 'Επιλογή τύπου ListBox
    ChartTypeComboBox.AddItem "3D-Columns" 'Προσθήκη τιμών σε ListBox
    ChartTypeComboBox.AddItem "3D-Bars"
    ChartTypeComboBox.AddItem "3D-Pie"
    ChartTypeComboBox.Value = ChartTypeComboBox.List(0) 'Προκαθορ. Τιμή
End Sub
```

Δημιουργία Γραφημάτων (Charts) με VBA (4/4)




```
' Επιστρέφει την κατάλληλα Σταθερά του Excel, που αντιστοιχεί
' στον επιλεγέντα, από τον χρήστη, «Τύπο Γραφήματος»
Private Function FindChartType(UserSelection As String)
    If UserSelection = "3D-Columns" Then
        FindChartType = xl3DColumn
    ElseIf UserSelection = "3D-Bars" Then
        FindChartType = xl3DBarClustered
    Else
        FindChartType = xl3DPie
    End If
End Function
```

```
' Δημιουργία του κατάλληλου Chart με βάση τις επιλογές του Χρήστη
Private Sub CreateChartButton_Click()     ' Για το κουμπί «Δημιουργία Γραφήματος»
    Dim rngChtData As Range, NameOfCurrentSheet As String

    On Error Resume Next

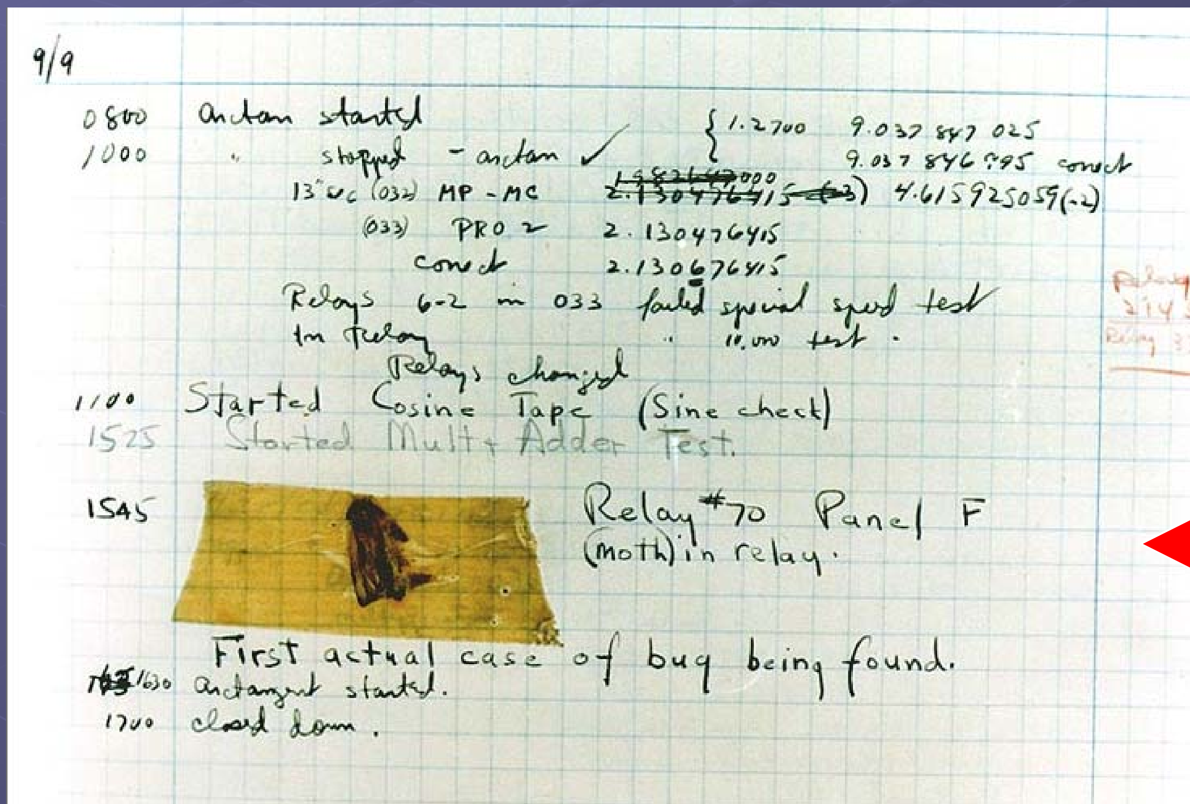
    NameOfCurrentSheet = ActiveSheet.Name    ' Το όνομα του Ενεργού Φύλλου
    Set rngChtData = Selection               ' Η Επιλεγμένη  περιοχή (Range)

    Charts.Add                               ' Προσθήκη αντικειμένου Chart
    With ActiveChart                         ' Καθορισμός Ιδιοτήτων του Chart
        .ChartType = FindChartType(ChartTypeComboBox.Value) ' Τύπος Chart
        .HasTitle = True                     ' Chart με Τίτλο
        .ChartTitle.Text = ChartTitleTextBox.Value   ' Καθορισμός Τίτλου
        .HasLegend = LegentCheckBox.Value    ' Ύπαρξη ή μη Υπομνήματος
        .Location Where := xlLocationAsObject, Name:=NameOfCurrentSheet
    End With
    ActiveChart.SetSourceData Source:=rngChtData ' Ορισμός Περιοχής Δεδομένων

    rngChtData.Select                   ' Αποκατάσταση της Επιλεγμένης Περιοχής
End Sub
```

Αποσφαλμάτωση (Debugging)

Το πρώτο Bug ήταν ένας σκώρος (moth) στον Υπολογιστή Mark II του Πανεπιστημίου Harvard, στις 9 Σεπτεμβρίου 1947. Μετά την αφαίρεση του (*debugging*), ο Υπολογιστής λειτούργησε και πάλι κανονικά.



Από το Βιβλίο Συμβάντων του Harvard της 9ης Σεπτ. 1947

Αποσφαλμάτωση (Debugging)

«Αποσφαλμάτωση» καλούμε τη διαδικασία εύρεσης και εξάλειψης σφαλμάτων από προγράμματα.

Η έννοια της «αποσφαλμάτωσης» είναι άρρηκτα συνδεδεμένη με τον Προγραμματισμό (Ανάπτυξη εφαρμογών).

Ο Bill Gates (ιδρυτής της Microsoft) είπε κάποτε:

> *«Εμείς στην Microsoft είμαστε κυρίως Testers (Δοκιμαστές), απλά έχουμε και λίγους Developers (Προγραμματιστές) για να μας δίνουν δουλειά».*

Όλα τα σύγχρονα Προγραμματιστικά Περιβάλλοντα, περιλαμβάνουν ειδικά εργαλεία και τεχνικές ώστε η αποσφαλμάτωση να γίνεται αποτελεσματικά, εύκολα και γρήγορα.

Αποσφαλμάτωση (Debugging) στη VBA

Το περιβάλλον της VBA, παρέχει έναν αριθμό από «*εργαλεία*» για την διευκόλυνση στην αποσφαλμάτωση των αναπτυσσόμενων προγραμμάτων.

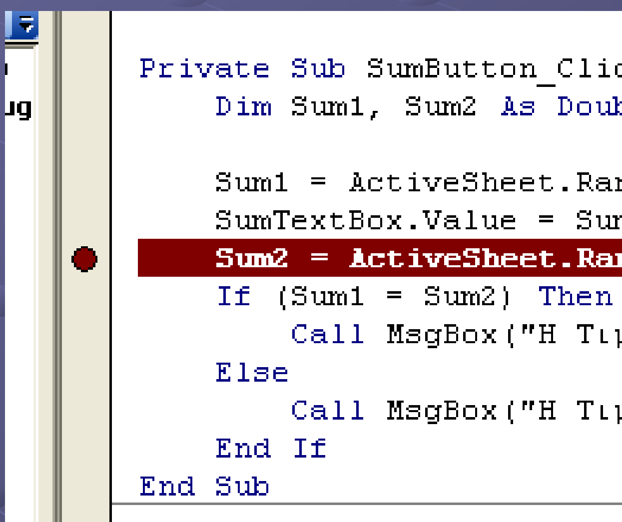
Σημαντικότερα από αυτά είναι:

- Η Δυνατότητα εκτέλεσης του Προγράμματος μας βήμα-βήμα
- Η Δυνατότητα προσθήκης *BreakPoints* (*Σημείων διακοπής*) στον Κώδικά μας
- Η Δυνατότητα ελέγχου των τιμών των Μεταβλητών του Προγράμματος μας σε κάθε βήμα εκτέλεσης του, μέσω των σχετικών παραθύρων (Watches, Locals, Immediate)

Αποσφαλμάτωση (Debugging) στη VBA
«*Break Points*»

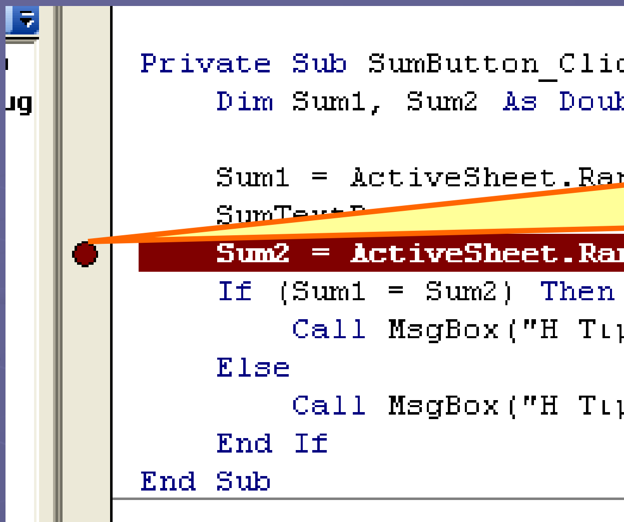
Ένα «*Break Point*» είναι ένα «*σημάδι*» σε κάποια γραμμή κώδικα, το οποίο λέει στην VBA να σταματήσει την εκτέλεση του προγράμματος ακριβώς πριν από την γραμμή κώδικα που έχει το «*Break point*».

Η εκτέλεση μπαίνει σε μια ειδική κατάσταση την “*Break Mode*”, η οποία μας επιτρέπει να βλέπουμε ή να αλλάζουμε τις τιμές των μεταβλητών, να εκτελούμε εντολές μέσω του παραθύρου «*Immediate*».

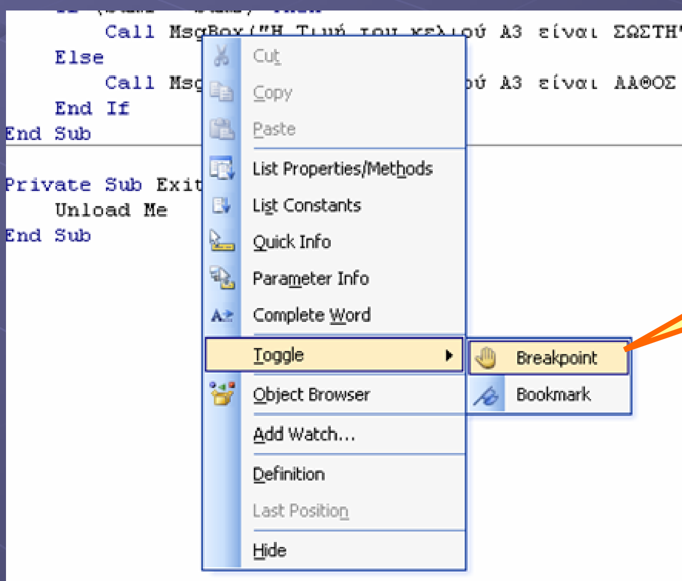


- Δεν μπορούμε να βάλουμε «*Break Point*» σε κενές γραμμές, ή γραμμές Δήλωσης Μεταβλητών (Dim).
- Μετά από ένα «*Break Point*» μπορούμε να συνεχίσουμε την εκτέλεση του κώδικα με F5, ή επιλέγοντας "*Continue*" από το μενού «*Run*», ή εκτελώντας τον κώδικα Βήμα-Βήμα.
- Τα «*Break Points*» δεν αποθηκεύονται μαζί με το WorkBook αλλά χάνονται όταν κλείσουμε το αρχείο μας.

Αποσφαλμάτωση (Debugging) στη VBA



Κάνοντας κλικ στο γκρι περιθώριο αριστερά από μία γραμμή κωδικα VBA, <u>τοποθετούμε</u> ή <u>αφαιρούμε</u> ένα ***BreakPoint*** στο σημείο αυτό.



Εναλλακτικά μπορούμε να κάνουμε δεξί κλικ πάνω στην γραμμή VBA και από το αναδυόμενο Menu να επιλέξουμε: ***Toggle → BreakPoint***.
Ακόμα μπορούμε να πατήσουμε «***F9***» ή να χρησιμοποιήσουμε το Menu «***Debug***»

Η εκτέλεση ενός προγράμματος VBA σταματάει μόλις φθάσει σε ένα ***BreakPoint***, οπότε μπορούμε να ελέγξουμε τις τιμές των μεταβλητών αυτή τη στιγμή και να εντοπίσουμε τυχόν σφάλματα στον κώδικά μας.

Αποσφαλμάτωση (Debugging) στη VBA

Εκτέλεση του κώδικα "*Βήμα-Βήμα*" (*Step Into / Over / Out*)

Μπορούμε να μπούμε σε κατάσταση εκτέλεσης «*Βήμα-Βήμα*» είτε πατώντας **F8** για την Διαδικασία (Sub) μέσα στην οποία βρίσκεται ο cursor είτε όταν σταματήσει η εκτέλεση του κώδικα σε κάποιο «*Break-Point*».
Στην κατάσταση «*Βήμα-Βήμα*», η επόμενη εντολή που πρόκειται να εκτελεστεί εμφανίζεται με κίτρινο φόντο.

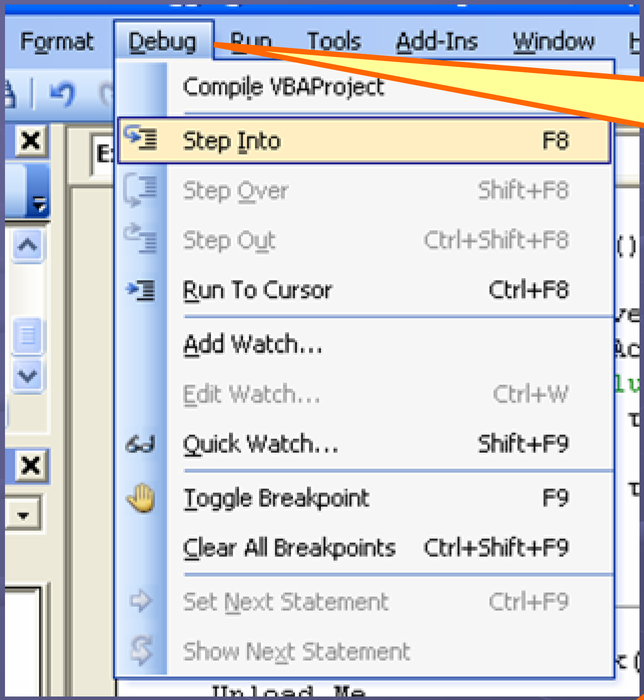
Εάν η Διαδικασία η οποία εκτελείται «*Βήμα-Βήμα*» καλεί μια άλλη Διαδικασία, θα εκτελεστεί και αυτή «*Βήμα-Βήμα*» εκτός αν πατήσουμε SHIFT+F8 που σημαίνει "*Step Over*" – δηλαδή η καλούμενη Διαδικασία δεν θα εκτελεστεί «*Βήμα-Βήμα*» αλλά θα εκτελεστεί σαν αν ήταν μία μοναδική εντολή.

Το πάτημα του CTRL+SHIFT+F8 την ώρα που εκτελείται κάποια Διαδικασία, σημαίνει "*Step Out*" – δηλαδή σταματά η κατάσταση εκτέλεσης *Βήμα-Βήμα* μέχρι το τέλος της εκτελούμενης Διαδικασίας και επανέρχεται αμέσως μετά την γραμμή κώδικα που είχε καλέσει την εν λόγω Διαδικασία.

Run To Cursor

Αυτό μας δίνει την δυνατότητα να ζητήσουμε από την VBA να εκτελέσει τον κώδικα μέχρι την γραμμή πάνω στην οποία βρίσκεται ο cursor. Ουσιαστικά ισοδυναμεί με ένα προσωρινό «*Break Point*».

Αποσφαλμάτωση (Debugging) στη VBA



Από το Μενού ***Debug*** έχουμε πρόσβαση στις περισσότερες από τις υπάρχουσες δυνατότητες «αποσφαλμάτωσης» της VBA

Τα παράθυρα ***Immediate***, ***Locals*** και ***Watches*** είναι ιδιαίτερα χρήσιμα στο να εντοπίζουμε τα σφάλματα του κώδικα, βλέποντας τις τιμές των Μεταβλητών μας ανά πάσα στιγμή.

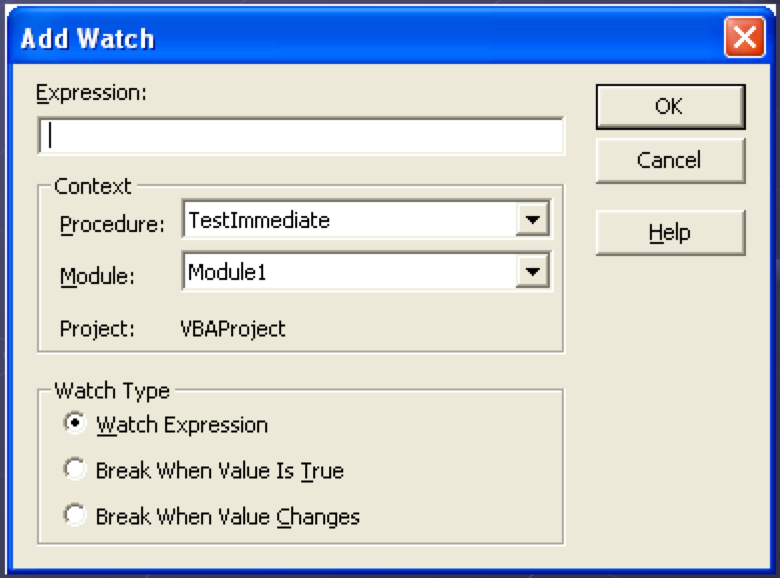


Αποσφαλμάτωση (Debugging) στη VBA
Παράθυρο “*Watch*”

Το παράθυρο «*Watch*» μας δίνει την δυνατότητα να παρακολουθούμε τις τιμές συγκεκριμένων μεταβλητών η εκφράσεων (*Watches*) καθώς και να σταματήσουμε την εκτέλεση μπαίνοντας σε «*Break mode*» είτε όταν η τιμή του «*Watch*» γίνει True είτε όταν η τιμή του «*Watch*» αλλάξει.

Μπορούμε να εμφανίσουμε το παράθυρο «*Watch*» από το μενού «*View*».

Για να προσθέσουμε μια Μεταβλητή ή Έκφραση στο παράθυρο «*Watch*», επιλέγουμε «*Add Watch*» από το μενού «*Debug*», οπότε θα εμφανιστεί το παράθυρο «*Add Watch*» που φαίνεται δίπλα.



Όπως βλέπουμε, στο πλαίσιο «*Watch Type*» υπάρχουν τρεις εναλλακτικοί τύποι Watches.

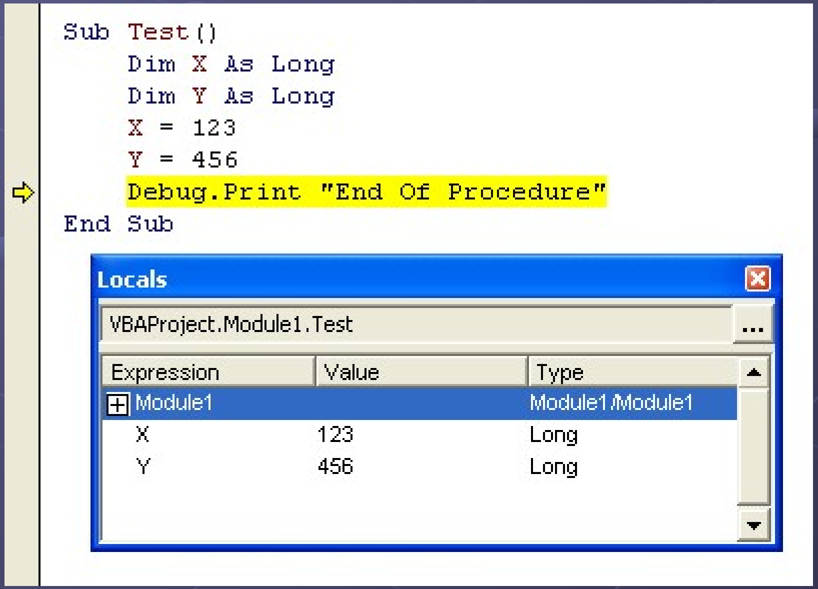
Μπορούμε να έχουμε πολλά ενεργά «*Watches*» ταυτόχρονα στο ίδιο Project.

Αποσφαλμάτωση (Debugging) στη VBA
Παράθυρο “*Locals*”

Το παράθυρο «*Locals*» εμφανίζει όλες τις (Τοπικές) μεταβλητές μιας Διαδικασίας καθώς και τις *Global* μεταβλητές με τις τιμές τους καθώς εκτελείται ο κώδικας *Βήμα-Βήμα*.

Δεν μπορούμε να αλλάξουμε τις τιμές των μεταβλητών μέσω του παραθύρου «*Locals*».

Μπορούμε να εμφανίσουμε το παράθυρο «*Locals*» από το μενού «*View*».



Αποσφαλμάτωση (Debugging) στη VBA
Παράθυρο "*Immediate*"

Το παράθυρο «*Immediate*» μας επιτρέπει να δώσουμε εντολές καθώς και να δούμε και να αλλάξουμε το περιεχόμενο των μεταβλητών είτε όταν βρισκόμαστε σε «*Break mode*» είτε όταν δεν εκτελείται κάποιος κώδικας. Μπορούμε να το εμφανίσουμε από το μενού «*View*» ή πατώντας CTRL+G.

Για παράδειγμα:

```
?ActiveCell.Address          ' Άμεση Εντολή (αρχίζει με ?)
$A$10                        ' Απάντηση (αποτέλεσμα εντολής)
```

Για να δώσουμε τιμή σε μια μεταβλητή δεν χρησιμοποιούμε το «?» στην αρχή, απλά πατάμε ENTER στο τέλος της εντολής.

Για παράδειγμα:

```
Range("A1").Value = "Ημερομηνία" <ENTER>
```

Για να εκτελέσουμε περισσότερες από μία εντολές, χρησιμοποιούμε τον χαρακτήρα «:» για να τις διαχωρίσουμε και τις εκτελούμε σαν μία «φυσική» εντολή.

Για παράδειγμα:

```
For j=1 To 10 : Debug.Print MyArray(j) : Next j <ENTER>
```

Αποσφαλμάτωση (Debugging) στη VBA

Debug.Print

Μπορούμε να χρησιμοποιήσουμε την εντολή «*Debug.Print*» οπουδήποτε στον κώδικά μας αν θέλουμε να εκτυπώσουμε την τρέχουσα τιμή κάποιας Μεταβλητής/Έκφρασης ή κάποιο μήνυμα στο παράθυρο «*Immediate*». Για παράδειγμα:

```
'
' Εντολές κώδικα VBA
'
Debug.Print   MyArray(4)   ' Εκτύπωση της τιμής του MyArray(4)
```

Debug.Assert

Η εντολή «*Debug.Assert (συνθήκη)* » σταματά την εκτέλεση του κώδικα εάν δεν ικανοποιείται η συνθήκη, όπου «*συνθήκη*» είναι μια έκφραση που επιστρέφει *True* ή *False*. Εάν η «*συνθήκη*» δίνει ως αποτέλεσμα το *False,* τότε η VBA μπαίνει σε «*Break Mode*» στο σημείο αυτό. Για παράδειγμα:

```
Dim i As Integer
i = 200
Debug.Assert (i < 90) ' Η συνθήκη είναι False, άρα η εκτέλεση
                      ' θα σταματήσει και θα μπει σε Break Mode
```

Αποσφαλμάτωση (Debugging) στην VBA

«*Κωδικοί Σφάλματος*» (Error Codes) της VBA (1/3)

VBA Error Codes	
Error code	Error message
3	Return without Gosub
5	Invalid procedure call
6	Overflow
7	Out of memory
9	Subscript out of range
10	Duplicate definition (versions 5.0 and 7.0)
10	This array is fixed or temporarily locked (version97)
11	Division by zero
13	Type mismatch
14	Out of string space
16	String formula too complex (versions 5.0 and 7.0)
16	Expression too complex (version 97)
17	Can't perform requested operation
18	User interrupt occurred
23	Resume without error
28	Out of stack space
35	Sub or function not defined (versions 5.0 and 7.0)
35	Sub, function, or property not defined (version 97)
47	Too many DLL application clients (version 97)
48	Error in loading DLL
49	Bad DLL calling convention
51	Internal error
52	Bad file name or number
53	File not found
54	Bad file mode
55	File already open

VBA Error Codes	
Error code	Error message
57	Device I/O error
58	File already exists
59	Bad record length
61	Disk full
61	Input past end of line
63	Bad record number
67	Too many files
68	Device unavailable
70	Permission denied
71	Disk not ready
74	Can't rename with different drive
75	Path/File access error
76	Path not found
91	Object variable not set (versions 5.0 and 7.0)
91	Object variable or with block variable not set (version 97)
92	For Loop not initialized
93	Invalid pattern string
94	Invalid use of Null
95	User-defined error (versions 5.0 and 7.0 only)
298	System DLL could not be loaded (version 97)
320	Can't use character device names in specified file names (version 97)
321	Invalid file format (version 97)
322	Can't create necessary temporary file (version 97)
323	Can't load module; invalid format (versions 5.0 and 7.0)
325	Invalid format in resource file (version 97)
327	Data value named was not found (version 97)

Αποσφαλμάτωση (Debugging) στην VBA

«*Κωδικοί Σφάλματος*» (Error Codes) της VBA (2/3)

VBA Error Codes	
Error code	Error message
328	Illegal parameter; can't write arrays (version 97)
335	Could not access system registry (version 97)
336	ActiveX component not correctly registered (version 97)
337	ActiveX component not found (version 97)
338	ActiveX component did not correctly run (version 97)
360	Object already loaded (version 97)
361	Can't load or unload this object (version 97)
363	Specified ActiveX control not found (version 97)
364	Object was unloaded (version 97)
365	Unable to unload within this context (version 97)
368	The specified file is out of date. This program requires a newer version (version 97)
371	The specified object can't be used as an owner form for Show (version 97)
380	Invalid property value (version 97)
381	Invalid property -array index (version 97)
382	Property Set can 't be executed at run time (version 97)
383	Property Set can't be used with a read-only property (version 97)
385	Need property-array index (version 97)
387	Property Set not permitted (version 97)
393	Property Get can't be executed at run time (version 97)
394	Property Get can't be executed on write-only property (version 97)
400	Form already displayed; can't show modally (version 97)
402	Code must close topmost modal form first (version 97)
419	Permission to use object denied (version 97)

VBA Error Codes	
Error code	Error message
422	Property not found (version 97)
423	Property or method not found
424	Object required
425	Invalid object use (version 97)
429	ActiveX component can't create object or return reference to this object (version 97)
430	Class doesn't support OLE Automation
430	Class doesn't support Automation (version 97)
432	File name or class name not found during Automation operation (version 97)
438	Object doesn't support this property or method
440	OLE Automation error
440	Automation error (version 97)
442	Connection to type library or object library for remote process has been lost (version 97)
443	Automation object doesn't have a default value (version 97)
445	Object doesn't support this action
446	Object doesn't support named arguments
447	Object doesn't support current locale settings
448	Named argument not found
449	Argument not optional
449	Argument not optional or invalid property assignment (version 97)
450	Wrong number of arguments
450	Wrong number of arguments or invalid property assignment (version 97)
451	Object not a collection

Αποσφαλμάτωση (Debugging) στην VBA

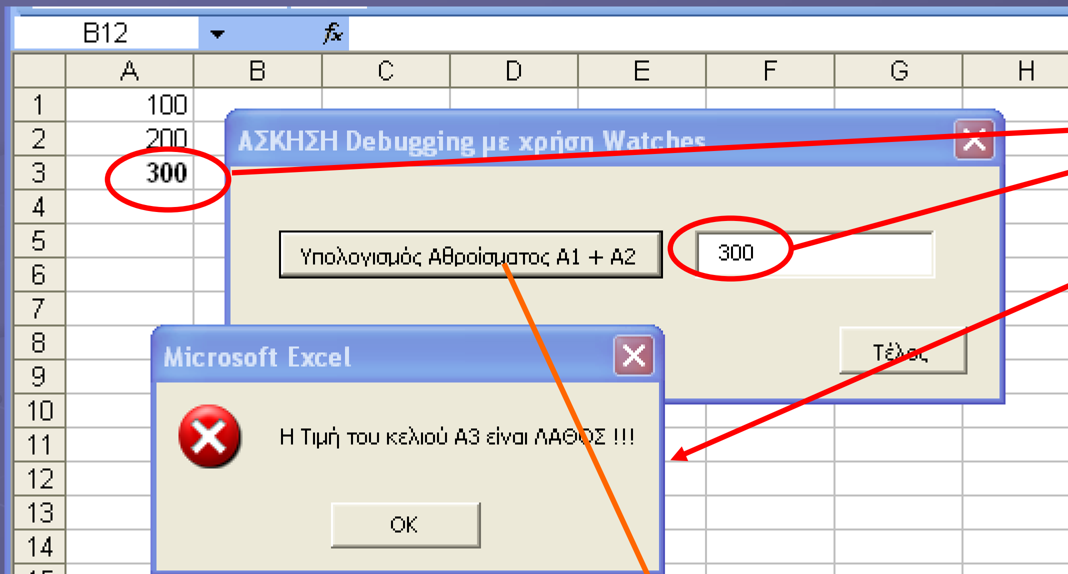
«*Κωδικοί Σφάλματος*» (Error Codes) της VBA (3/3)

VBA Error Codes	
Error code	**Error message**
452	Invalid ordinal
453	Specified DLL function not found
454	Code resource not found
455	Code resource lock error
457	This key is already associated with an element of this collection (version 97)
458	Variable uses a type not supported in Visual Basic (version 97)
459	This component doesn't support events (version 97)
460	Invalid clipboard format (version 97)
461	Specified format doesn't match format of data (version 97)
480	Can't create AutoRedraw image (version 97)
481	Invalid picture (version 97)
482	Printer error (version 97)
483	Printer driver does not support specified property (version 97)
484	Problem getting printer information from the system. Make sure the printer is set up correctly (version 97)
485	Invalid picture type (version 97)
486	Can't print form image to this type of printer (version 97)
735	Can't save file to Temp directory (version 97)
744	Search text not found (version 97)
746	Replacements too long (version 97)

VBA Error Codes	
Error code	**Error message**
1000	Classname does not have propertyname property (versions 5.0 and 7.0)
1001	Classname does not have methodname method (versions 5.0 and 7.0)
1002	Missing required argument argumentname (versions 5.0 and 7.0)
1003	Invalid number of arguments (versions 5.0 and 7.0)
1004	Methodname method of classname class failed (versions 5.0 and 7.0)
1005	Unable to set the propertyname property of the classname class (versions 5.0 and 7.0)
1006	Unable to get the propertyname property of the classname class (versions 5.0 and 7.0)
31001	Out of memory (version 97)
31004	No object (version 97)
31018	Class is not set (version 97)
31027	Unable to activate object (version 97)
31032	Unable to create embedded object (version 97)
31036	Error saving to file (version 97)
31037	Error loading from file (version 97)

Αποσφαλμάτωση (Debugging) στη VBA

Παράδειγμα χρήσης *Watches* (1/4)



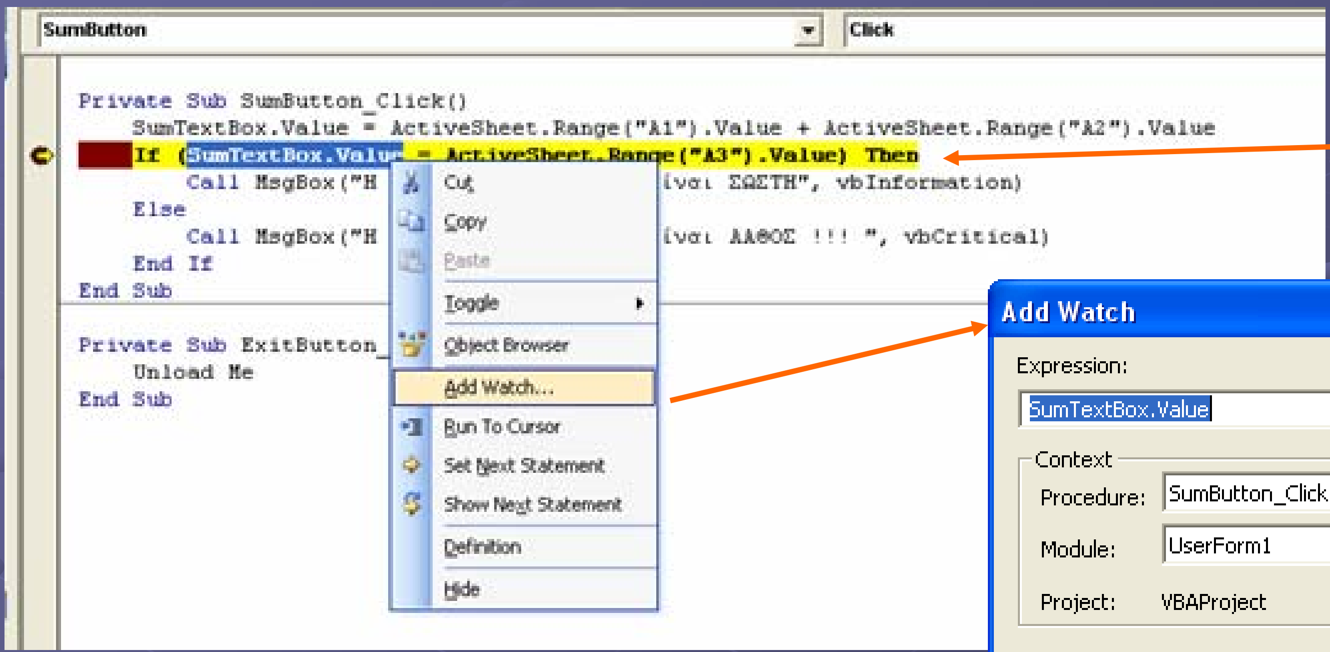
Παρότι και οι δύο τιμές είναι 300 εμφανίζεται μήνυμα Λάθους.
Για τον εντοπισμό θα χρησιμοποιήσουμε Watches

```
Private Sub SumButton_Click()
    SumTextBox.Value = ActiveSheet.Range("A1").Value + ActiveSheet.Range("A2").Value
    If (SumTextBox.Value = ActiveSheet.Range("A3").Value) Then
        Call MsgBox("Η Τιμή του κελιού Α3 είναι ΣΩΣΤΗ", vbInformation)
    Else
        Call MsgBox("Η Τιμή του κελιού Α3 είναι ΛΑΘΟΣ !!! ", vbCritical)
    End If
End Sub

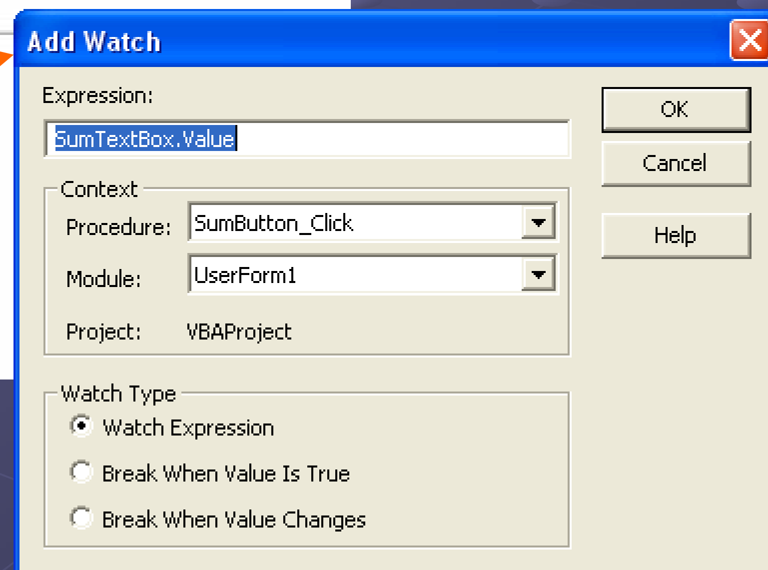
Private Sub ExitButton_Click()
    Unload Me
End Sub
```

Αποσφαλμάτωση (Debugging) στη VBA

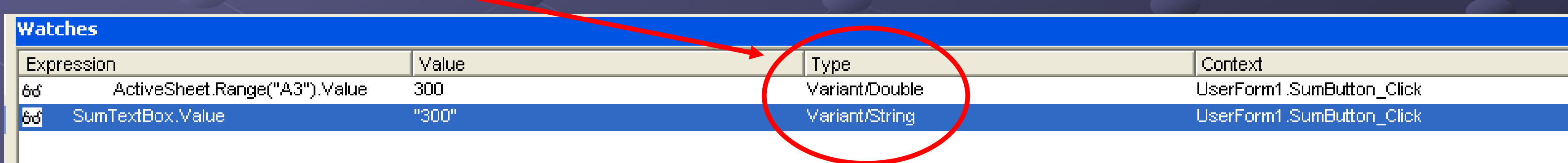
Παράδειγμα χρήσης *Watches* (2/4)



Βάζουμε ένα BreakPoint

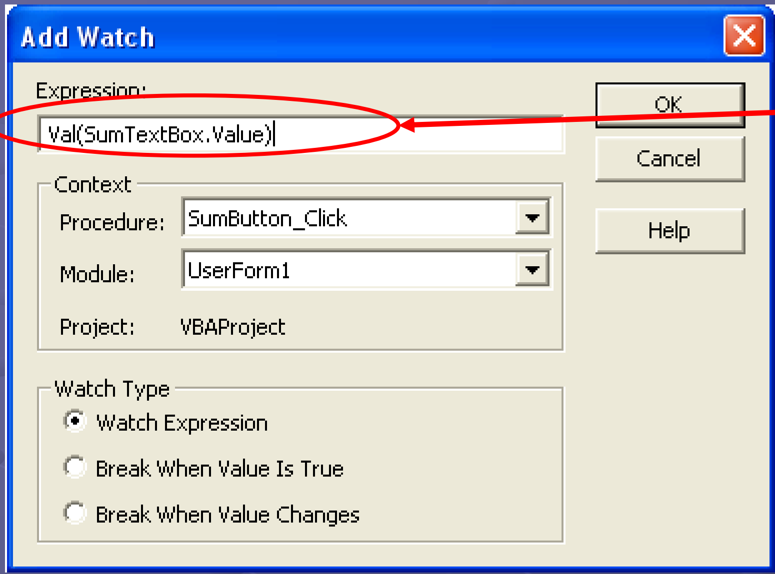


Προσθέτουμε 2 Watches και βλέπουμε ότι διαφέρει ο Τύπος των Μεταβλητών.



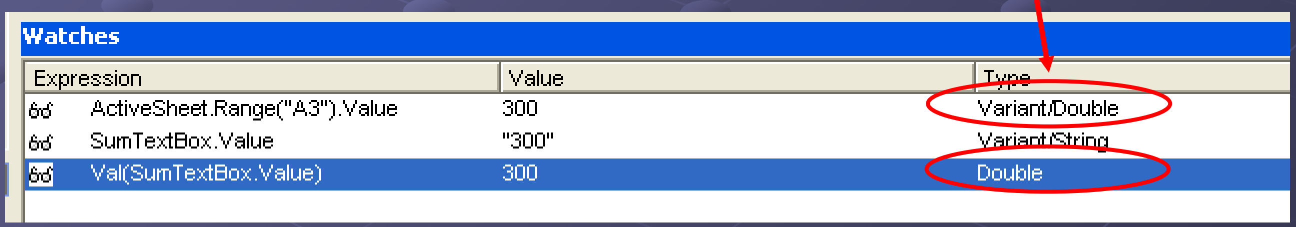
Αποσφαλμάτωση (Debugging) στη VBA

Παράδειγμα χρήσης *Watches* (3/4)



Προσθέτουμε ένα Watch για να δοκιμάσουμε τι θα γίνει αν χρησιμοποιήσουμε την Συνάρτηση Val().

Παρατηρούμε ότι τώρα οι Μεταβλητές έχουν τον ίδιο Τύπο (Double).



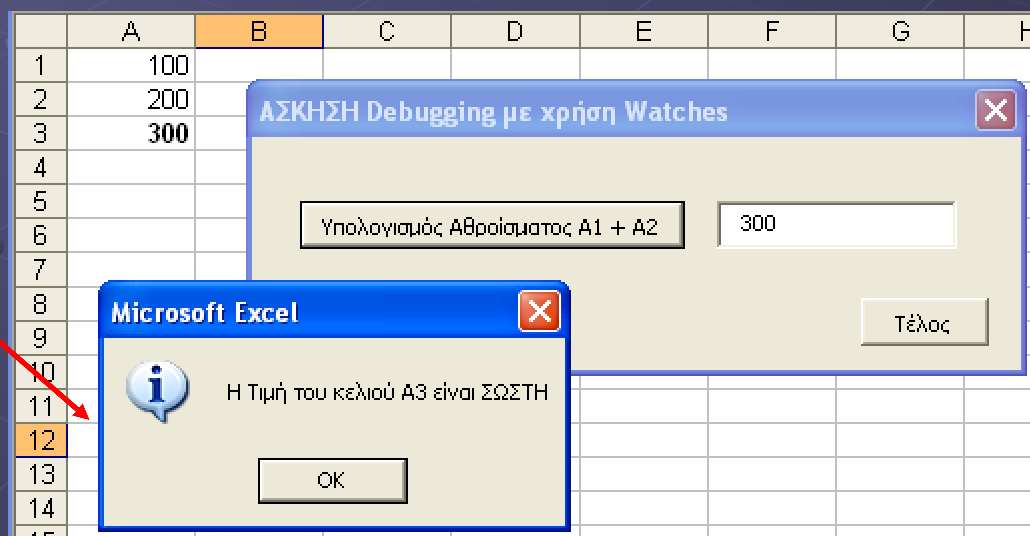
Αποσφαλμάτωση (Debugging) στη VBA

Παράδειγμα χρήσης *Watches* (4/4)

```
Private Sub SumButton_Click()
    SumTextBox.Value = ActiveSheet.Range("A1").Value + ActiveSheet.Range("A2").Value
    If (Val(SumTextBox.Value) = ActiveSheet.Range("A3").Value) Then
        Call MsgBox("Η Τιμή του κελιού Α3 είναι ΣΩΣΤΗ", vbInformation)
    Else
        Call MsgBox("Η Τιμή του κελιού Α3 είναι ΛΑΘΟΣ !!! ", vbCritical)
    End If
End Sub

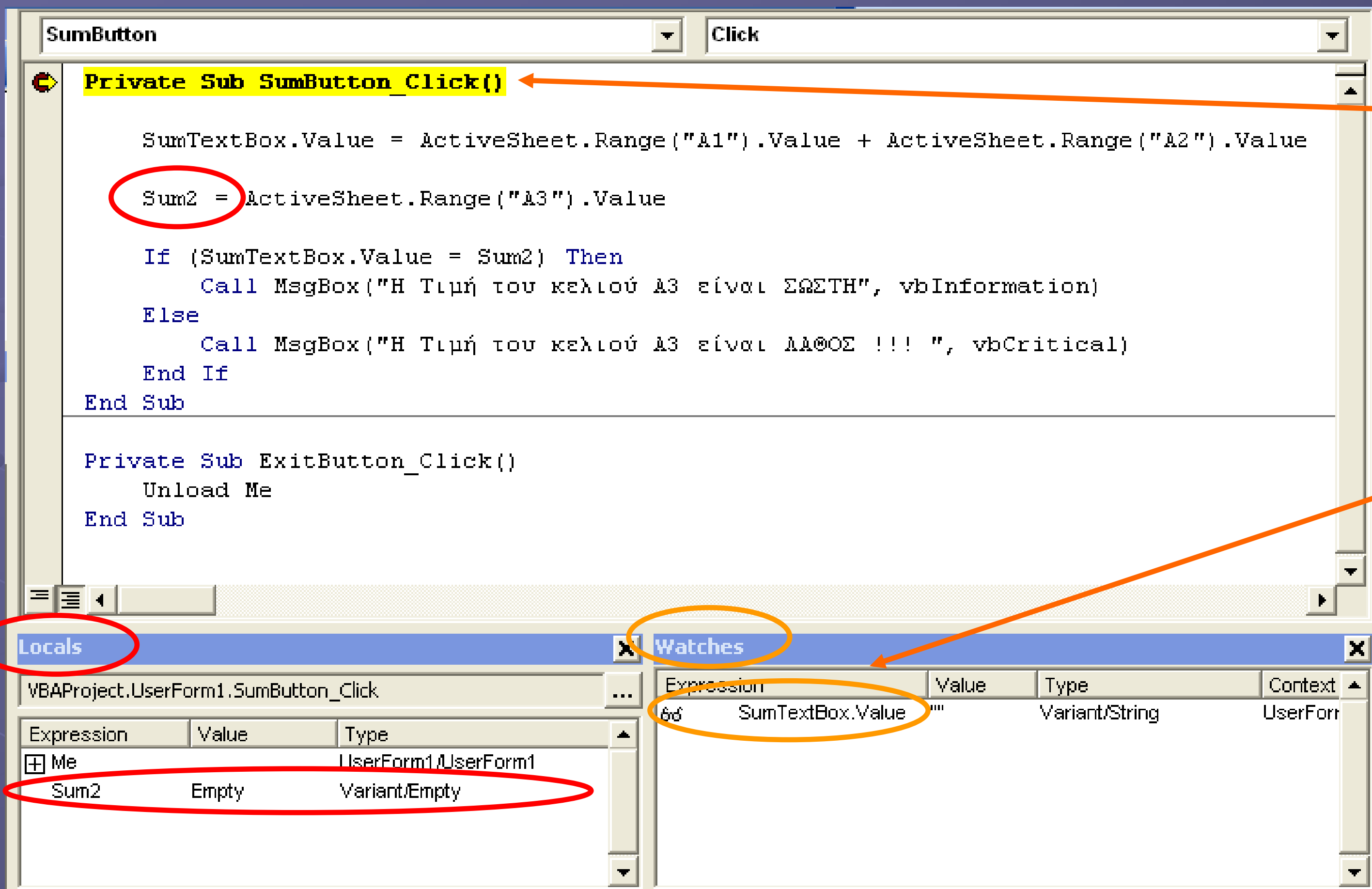
Private Sub ExitButton_Click()
    Unload Me
End Sub
```

Έχοντας πλέον εντοπίσει το πρόβλημα, κάνουμε την διόρθωση του κώδικα, ο οποίος τώρα λειτουργεί κανονικά.



Αποσφαλμάτωση (Debugging) στη VBA

Παράδειγμα συνδυασμού *Locals & Watches* (1/3)



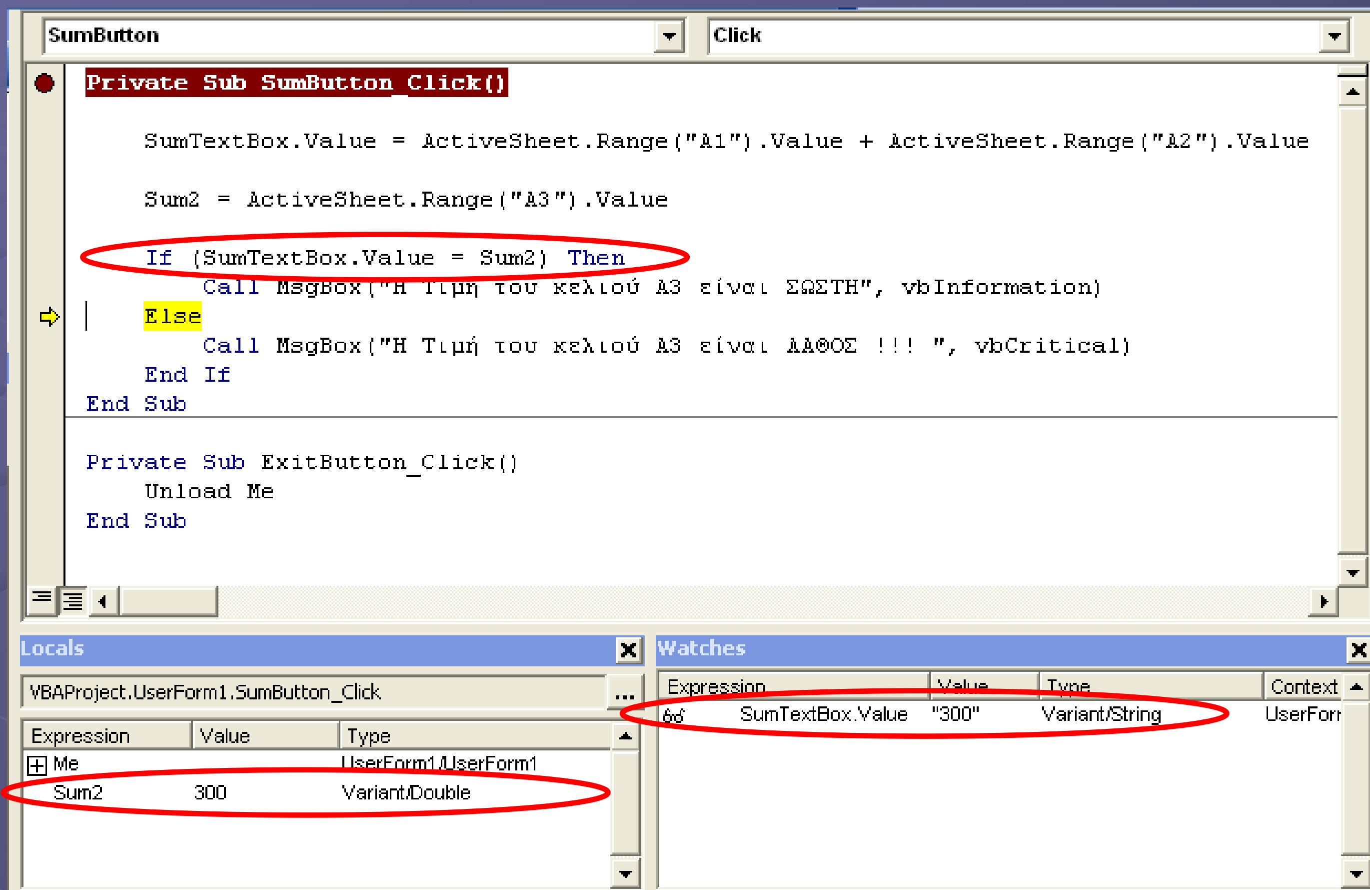
Βάζουμε ένα *BreakPoint* και χρησιμοποιούμε το ***F8*** για Βηματική Εκτέλεση (*Step Into*)

Ορίζουμε ένα *Watch* στην έκφραση *SumTextBox.Value*

Εμφανίζουμε το Παράθυρο *Locals* (από το μενού View), όπου φαίνεται η Τοπική Μεταβλητή *Sum2* (*Τιμή* και *Τύπος*).

Αποσφαλμάτωση (Debugging) στη VBA

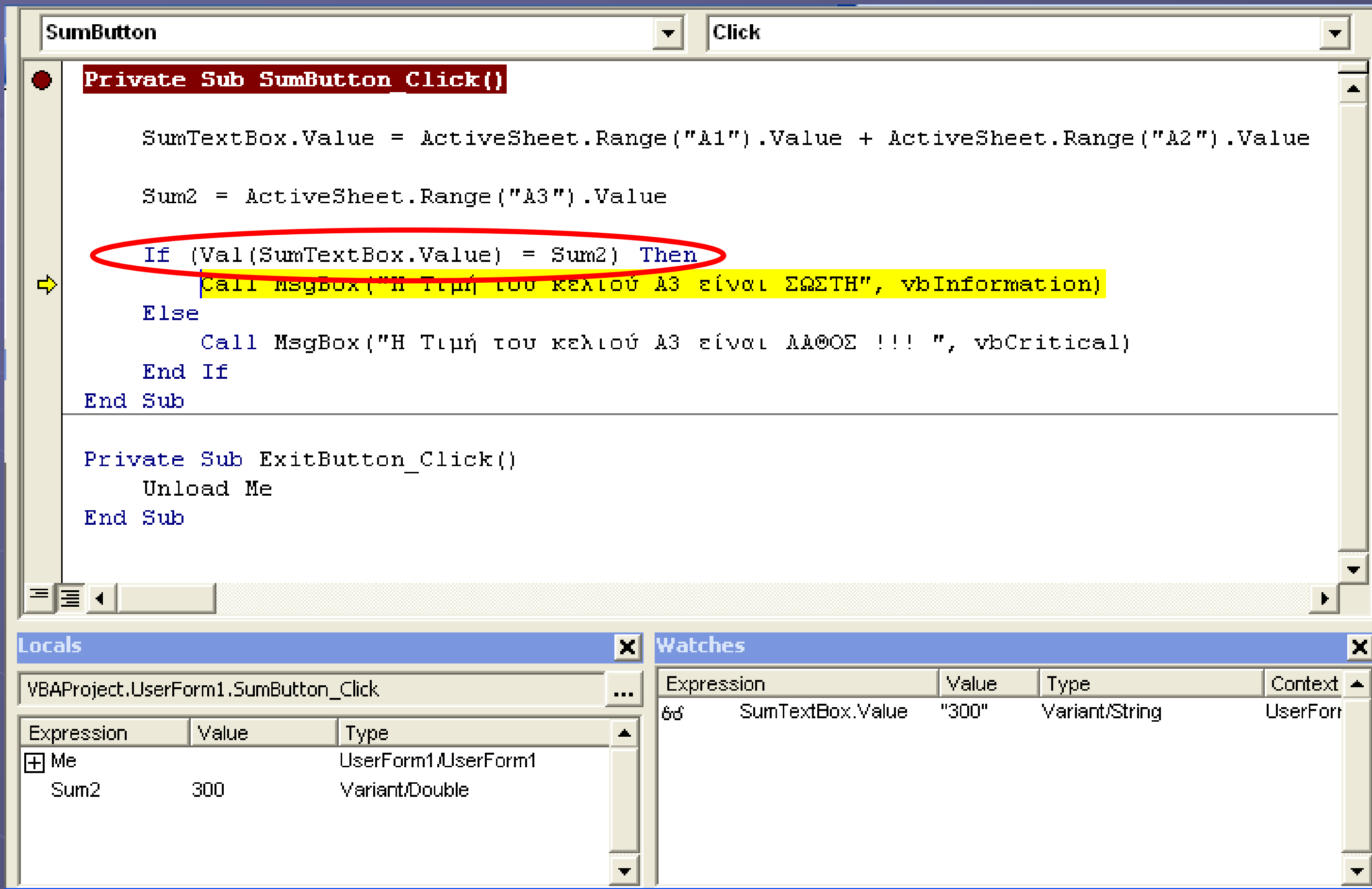
Παράδειγμα συνδυασμού *Locals & Watches* (2/3)



Παρατηρούμε ότι η Συνθήκη (*SumTextBox.Value = Sum2*) είναι *False* γιατί τα δύο μέρη έχουν διαφορετικό *Τύπο*.

Αποσφαλμάτωση (Debugging) στη VBA

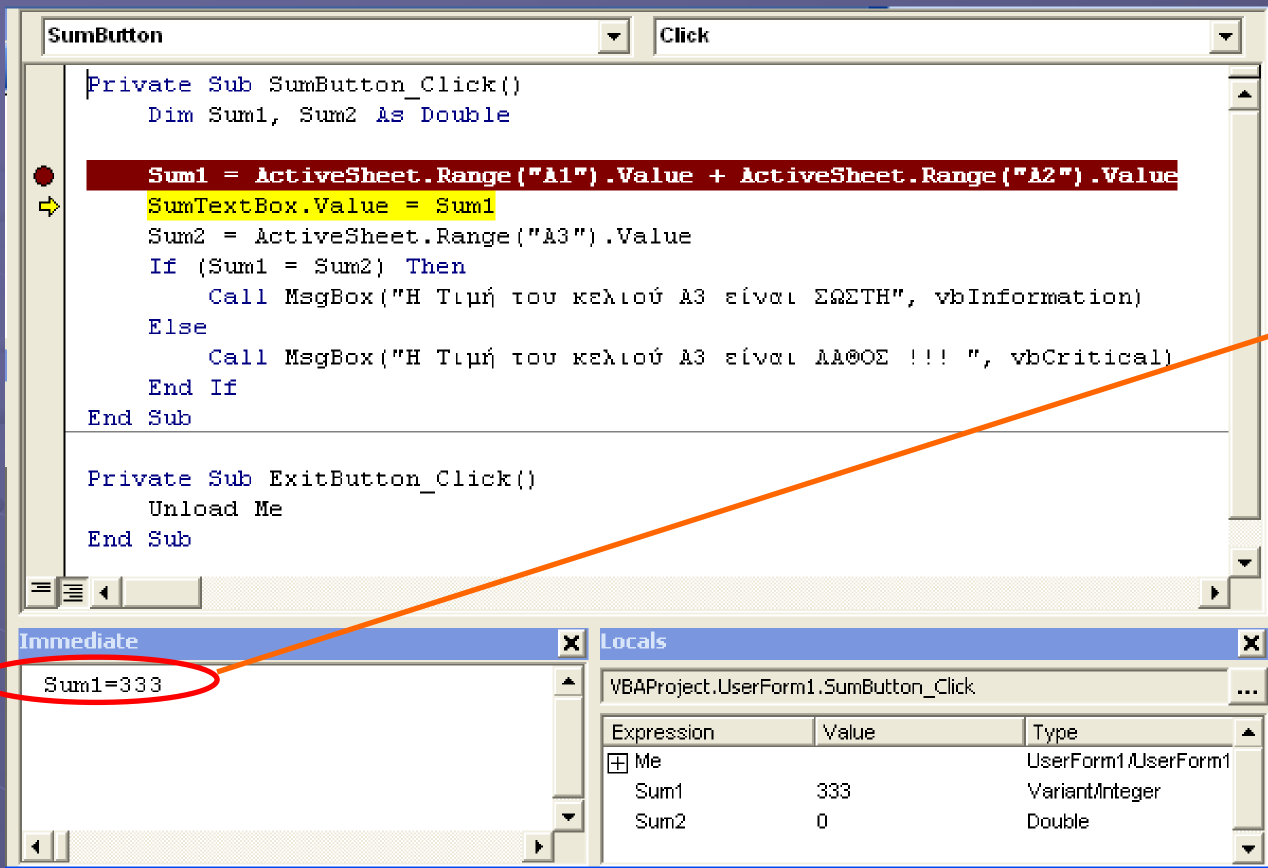
Παράδειγμα συνδυασμού *Locals & Watches* (3/3)



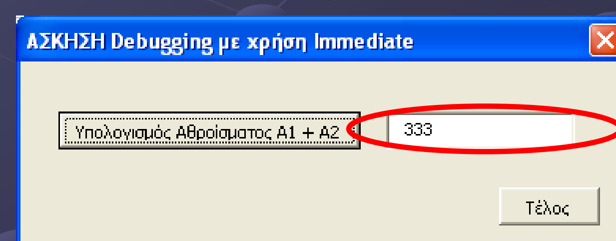
Διορθώνουμε τον κώδικα μας, χρησιμοποιώντας την Συνάρτηση *Val()* (η οποία θα μετατρέψει το String σε αριθμό) και παρατηρούμε ότι τώρα λειτουργεί κανονικά.

Αποσφαλμάτωση (Debugging) στη VBA

Παράδειγμα χρήσης *Immediate* (1/2)

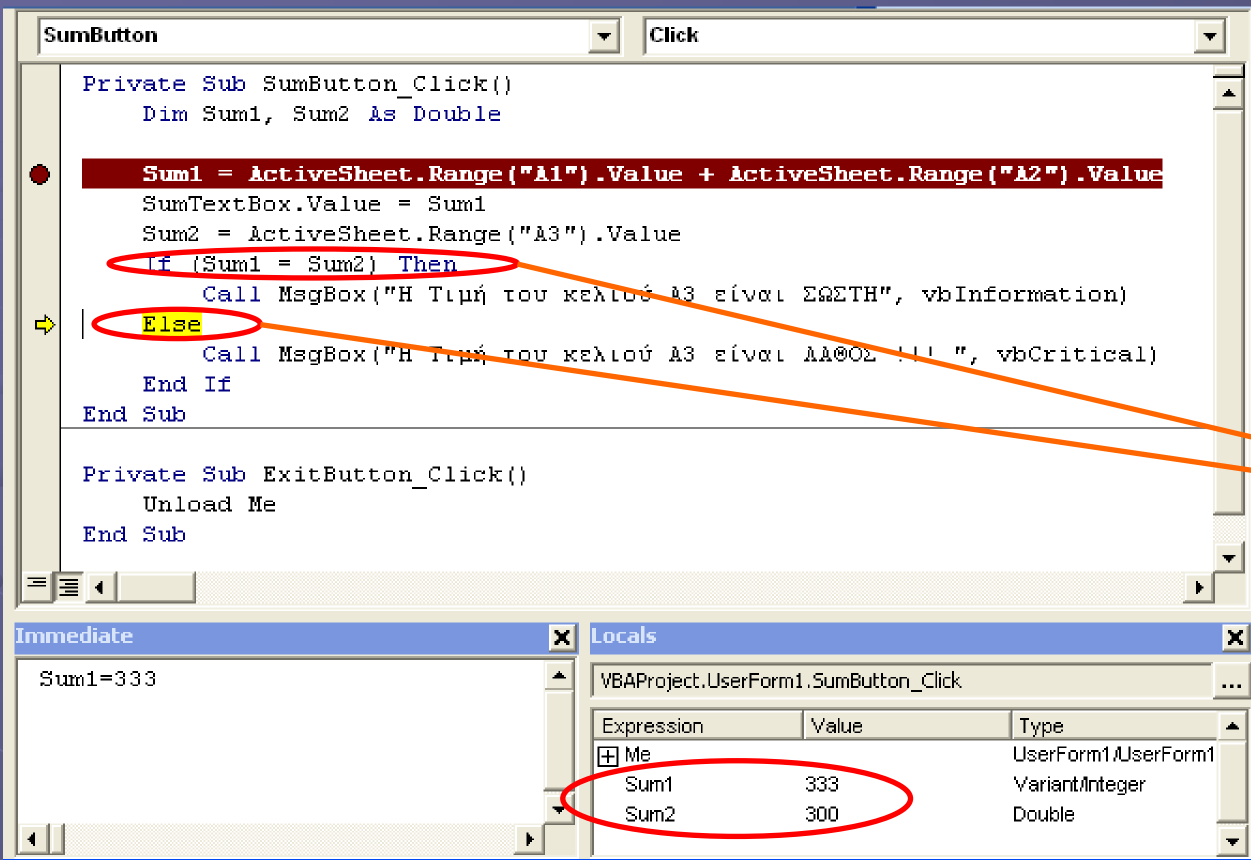


Στο παράθυρο *Immediate* αλλάζουμε την ώρα της εκτέλεσης του κώδικα την τιμή της μεταβλητής *Sum1* σε μία αυθαίρετη τιμή (εδώ 333). Η νέα αυτή τιμή ισχύει κανονικά πλέον.

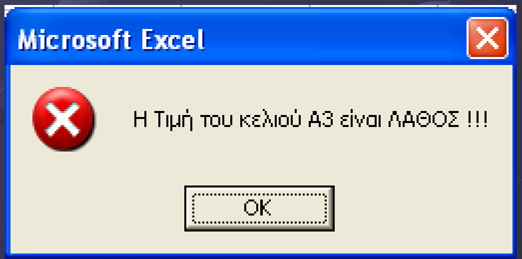


Αποσφαλμάτωση (Debugging) στη VBA

Παράδειγμα χρήσης *Immediate* (2/2)



Λόγω της τιμής που δώσαμε στην μεταβλητή *Sum1* η συνθήκη: *Sum1 = Sum2* είναι *False*.



Διαχείριση Σφαλμάτων στη VBA
Βασικά “*Εργαλεία* ”

Η εντολή On Error

Η εντολή «*On Error*» λέει στην VBA τι πρέπει να κάνει όταν συμβεί κάποιο σφάλμα. Επίσης μας δίνει την δυνατότητα να ενεργοποιήσουμε ή να απενεργοποιήσουμε την «παγίδευση» των Σφαλμάτων (Error trapping).

Η εντολή Resume

Η εντολή «*Resume*» λέει στην VBA από ποιο σημείο του κώδικα να συνεχίσει την εκτέλεση του προγράμματος, ύστερα από κάποια διακοπή που έγινε - συνήθως από «παγίδευση» ενός Σφάλματος μέσω της εντολής «*On Error*».

Το αντικείμενο Err

Το αντικείμενο «*Err*» περιέχει όλα τα στοιχεία ενός Σφάλματος (Error) – π.χ. το *Err.Number* περιέχει τον αριθμό Σφάλματος και το *Err.Description* την περιγραφή του. Επιπλέον έχει τις μεθδους *Clear* για αρχικοποίηση και *Raise* για την δημιουργία «τεχνητού» σφάλματος.

Η συνάρτηση Error()

Η συνάρτηση «*Error()*» επιστρέφει την περιγραφή που αντιστοιχεί σε έναν δεδομένο αριθμό Σφάλματος, τον οποίο της περνάμε ως όρισμα.

Διαχείριση Σφαλμάτων στη VBA
Η εντολή “*On Error*” (1/3)

Η σημαντικότερη εντολή για την διαχείριση σφαλμάτων (Error Handling) στην VBA είναι η εντολή «*On Error*». Με την εντολή αυτή λέμε στην VBA τι να κάνει όταν συμβεί κάποιο σφάλμα κατά την εκτέλεση του προγράμματος μας (Run time error).
Υπάρχουν τρεις μορφές της εντολής «*On Error*»:

1) On Error Goto 0
2) On Error Resume Next
3) On Error Goto <label>:

Ο πρώτος τρόπος, «On Error Goto 0», είναι ο προκαθορισμένος (*Default*) στην VBA. Με την εντολή αυτή, λέμε στην VBA να εμφανίσει, κάθε φορά που εμφανίζεται ένα λάθος κατά την εκτέλεση του προγράμματος, το standard Message Box, το οποίο μας δίνει την δυνατότητα είτε να μεταφερθούμε σε κατάσταση εκσφαλμάτωσης (*Debug mode*) είτε να τερματίσουμε το πρόγραμμα.

Ουσιαστικά η εντολή «On Error Goto 0» ισοδυναμεί με απενεργοποίηση του Διαχειριστή Σφαλμάτων (Error Handler).

Διαχείριση Σφαλμάτων στη VBA
Η εντολή "*On Error*" (2/3)

Με τον δεύτερο τρόπο «On Error Resume Next», λέμε στη VBA να αγνοήσει το Σφάλμα και να συνεχίσει την εκτέλεση του προγράμματος από την επόμενη εντολή.

Όμως η εντολή «On Error Resume Next», σε καμία περίπτωση δεν διορθώνει το Σφάλμα που προέκυψε. Είναι επομένως ευθύνη του Προγραμματιστή να ελέγξει για τυχόν Σφάλματα (errors) και να τα διαχειριστεί με τον κατάλληλο τρόπο.
Μπορούμε να ελέγξουμε αν υπάρχει Σφάλμα, μέσω της μεταβλητής «Err.Number». Αν αυτή δεν έχει τιμή 0, τότε υπάρχει Σφάλμα το οποίο ανάλογα με τον Κωδικό του (που δείχνει ποιο σφάλμα είναι) πρέπει να το αντιμετωπίσουμε με τον κατάλληλο κώδικα.

Για παράδειγμα:

```
On Error Resume Next
 N = 1 / 0                    ' Πρόκληση Σφάλματος
 If Err.Number <> 0 Then ' Έλεγχος για ύπαρξη Σφάλματος
     N = 1             ' Κώδικας για την Διόρθωση του Σφάλματος
 End If
```

Ο ανωτέρω κώδικας θα προκαλέσει το Σφάλμα #11 (*Division By Zero*). Επειδή όμως υπάρχει η εντολή On Error Resume Next , η εκτέλεση θα συνεχιστεί με την επόμενη εντολή, δηλαδή την *If Err.Number …….End If*, η οποία περιέχει τον κατάλληλο κώδικα που διορθώνει το Σφάλμα.

Διαχείριση Σφαλμάτων στη VBA
Η εντολή “*On Error*” (3/3)

Ο δεύτερος τρόπος «On Error Goto <label>:» λέει στην VBA να συνεχίσει την εκτέλεση του προγράμματος με την εντολή που ακολουθεί την προκαθορισμένη *<label>*.

Για παράδειγμα:

```
    On Error Goto ErrHandler:
    N = 1 / 0     ' Πρόκληση Σφάλματος
    '
    ' άλλες εντολές
    '
    Exit Sub

  ErrHandler:
        ' Εντολές για την διαχείριση του Σφάλματος
    Resume Next
End Sub
```

Διαχείριση Σφαλμάτων στη VBA
Η εντολή “*Resume*” (1/3)

Η εντολή «Resume» όταν χρησιμοποιείται μόνη της, λέει στην VBA να συνεχίσει την εκτέλεση του προγράμματος από την γραμμή του κώδικα που προκάλεσε το Σφάλμα.

Είναι επομένως σημαντικό να βεβαιωθούμε ότι έχουμε διορθώσει πρώτα το πρόβλημα (μέσω του Κώδικα Διαχείρισης Σφάλματος – Error Handling Code), διαφορετικά θα δημιουργηθεί ένας ατέρμων βρόχος (endless loop) μεταξύ της γραμμής που προκάλεσε το Σφάλμα και του Κώδικα Διαχείρισης Σφάλματος (*Error Handling Block*).

Στο παρακάτω παράδειγμα γίνεται απόπειρα ενεργοποίησης Φύλλο Εργασίας που δεν υπάρχει και διαχείριση του Σφάλματος που προκαλείται:

```
    On Error GoTo ErrHandler:
    Worksheets("NewSheet").Activate ' Πρόκληση Σφάλματος
    Exit Sub
  ErrHandler:
    If Err.Number = 9 Then           ' Έλεγχος για ύπαρξη Σφάλματος
        Worksheets.Add.Name = "NewSheet" ' Διόρθωση Σφάλματος
        Resume     ' Επιστροφή στην εντολή που προκάλεσε το Σφάλμα
    End If
```

Διαχείριση Σφαλμάτων στη VBA
Η εντολή “*Resume*” (2/3)

Ο δεύτερος τρόπος χρήσης της εντολής Resume είναι το «Resume Next». Η εντολή αυτή λέει στην VBA να συνεχίσει την εκτέλεση του προγράμματος με την εντολή που ακολουθεί αυτήν που προκάλεσε το σφάλμα.

Στο παρακάτω παράδειγμα προκαλείται Σφάλμα (Σφάλμα #11 - Division By Zero). Στον Κώδικα Διαχείρισης Σφάλματος (*Error Handling Block*) αντιμετωπίζεται το πρόβλημα που δημιούργησε το Σφάλμα (error) και κατόπιν συνεχίζεται η εκτέλεση από την επόμενη εντολή από αυτήν που προκάλεσε το Σφάλμα:

```
    On Error GoTo ErrHandler:
    N = 1 / 0          ' Πρόκληση Σφάλματος
    Debug.Print N
    Exit Sub

  ErrHandler:
    N = 1              ' Διόρθωση Σφάλματος
    Resume Next        ' Επιστροφή στην επόμενη εντολή από αυτήν που
                       ' προκάλεσε το Σφάλμα
```

Διαχείριση Σφαλμάτων στη VBA
Η εντολή “*Resume*” (3/3)

Ο τρίτος τρόπος χρήσης της εντολής Resume είναι Resume <label>: . Η εντολή αυτή λέει στην VBA να συνεχίσει την εκτέλεση του προγράμματος στο σημείο <label>: . Με τον τρόπο αυτό μπορούμε να παραλείψουμε ένα κομμάτι κώδικα αν συμβεί κάποιο Σφάλμα.

Για παράδειγμα:

```
    On Error GoTo ErrHandler:
    N = 1 / 0                      ' Πρόκληση Σφάλματος
    '
    ' Κώδικας που παραλείπεται αν συμβεί Σφάλμα
    '
  Label1:
    '
    ' άλλες εντολές
    '
    Exit Sub
  ErrHandler:
        Resume Label1:     ' Επιστροφή στην γραμμή «Label1:»
```

Η εντολή *Resume* όταν βρίσκεται μέσα στο Μπλοκ Διαχείρισης Σφάλματος (Error Handling Block), αρχικοποιεί (*Clear*) το αντικείμενο Err.

Διαχείριση Σφαλμάτων στη VBA
Το Αντικείμενο "*Err* "

Το αντικείμενο «*Err*» περιέχει όλα τα στοιχεία ενός Σφάλματος (*Error*) καθώς και μεθόδους για την *αρχικοποίηση* του και για την δημιουργία «τεχνητού» σφάλματος.

Σημαντικές *Ιδιότητες* (Properties):

- Number ‘ Ο Αριθμός του Σφάλματος (Error Code)
- Description ‘ Η Περιγραφή (λεκτικό) του Σφάλματος
- Source ‘ Το όνομα της Εφαρμογής ή του Αντικειμένου που το προκάλεσε

Σημαντικές *Μέθοδοι* (Methods):

- Clear ‘ Για την αρχικοποίηση (καθαρισμό) του αντικειμένου Err
- Raise ‘ Για την δημιουργία «τεχνητού» (user-defined) Σφάλματος

Παραδείγματα χρήσης του αντικειμένου Err:

```
                                ' Εμφάνιση του Λεκτικού του Σφάλματος
MsgBox "Συνέβη το εξής σφάλμα: " & Err.Description
Err.Clear                       ' Αρχικοποίηση του αντικειμένου Err
Err.Raise 2000                  ' Δημιουργία Σφάλματος με Αριθμό 2000
```

Διαχείριση Σφαλμάτων στη VBA

Η Συνάρτηση “*Error()* ”

Η συνάρτηση Error() επιστρέφει την περιγραφή που αντιστοιχεί σε έναν δεδομένο αριθμό Σφάλματος.

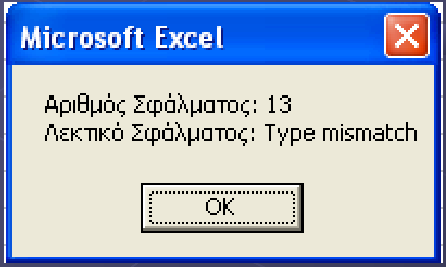
Τρόπος σύνταξης: Error(*errornumber*)

όπου *errornumber* ο Αριθμός Σφάλματος. Αν δεν δοθεί το *errornumber* επιστρέφει την περιγραφή του πιο πρόσφατου Σφάλματος.

Για Παράδειγμα, αν έχει συμβεί το Σφάλμα #13, η παρακάτω εντολή:

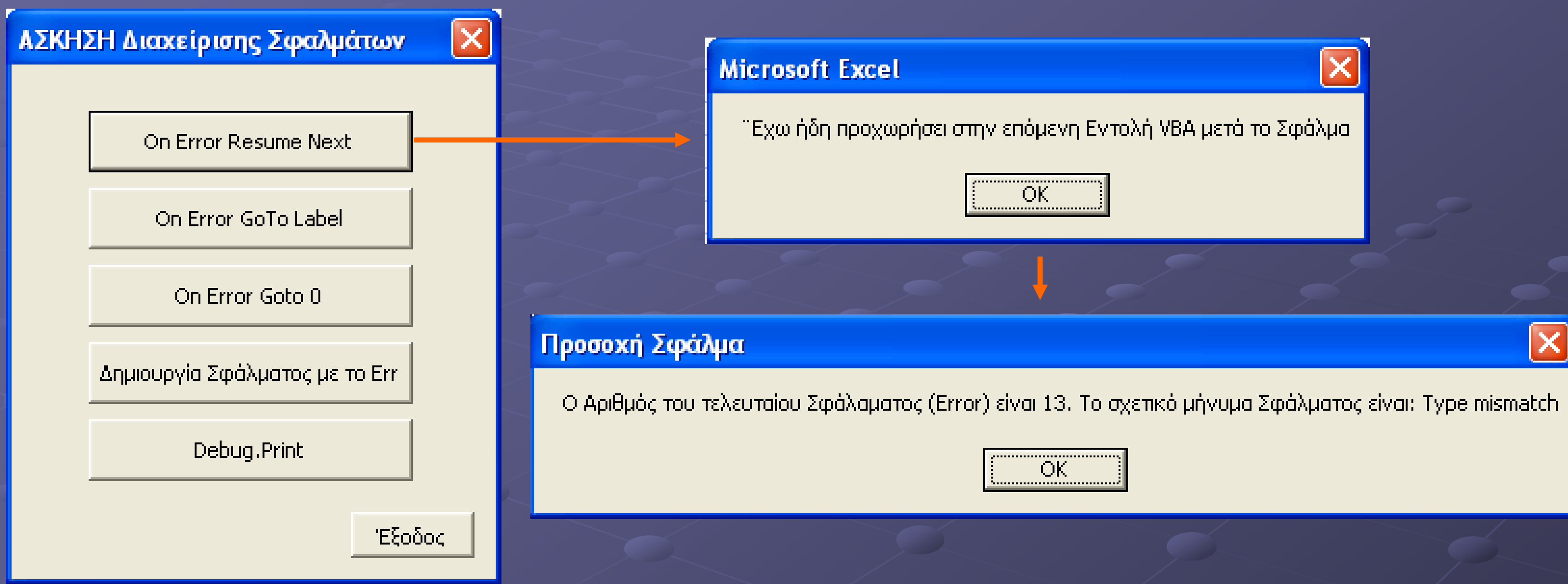
```
Msg = "Αριθμός Σφάλματος: " & Error(Err.Number) & vbCrLf & _
        "Λεκτικό Σφάλαματος: " & Error(Err.Number)
```

Θα εμφανίζει το εξής:



Διαχείριση Σφαλμάτων στη VBA

Παράδειγμα χρήσης *OnError, Resume, Err, Error(), Debug.Print* (1/5)



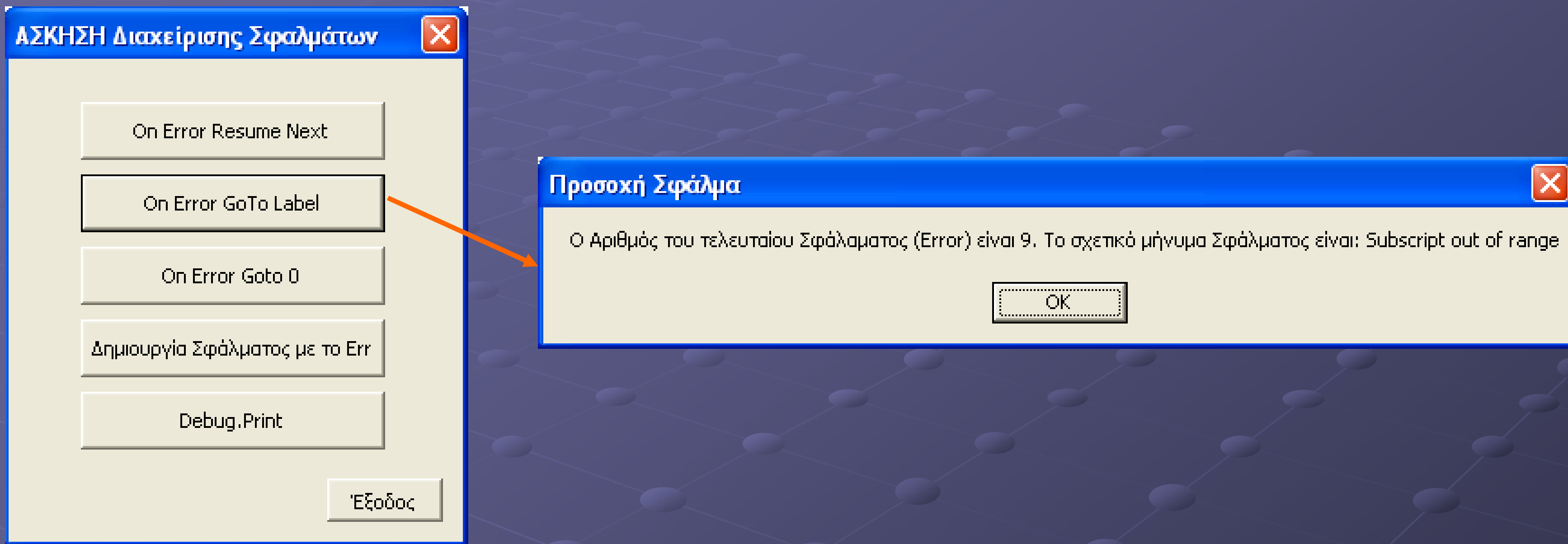

```
Private Sub GotoNextButton_Click()
    Dim i As Integer, Msg As String

    On Error Resume Next
    i = "String Test"    ' Προσπάθεια αποθήκευσης String σε Ακέραια Μεταβλητή

    MsgBox "¨Εχω ήδη προχωρήσει στην επόμενη Εντολή VBA μετά το Σφάλμα"
    Msg = "Ο Αριθμός του τελευταίου Σφάλαματος (Error) είναι " & Err.Number & _
      ". Το σχετικό μήνυμα Σφάλματος είναι: " & Error(Err.Number)  ' Χρήση της συνάρτησης Error()
    MsgBox Msg, , "Προσοχή Σφάλμα"
End Sub
```

Διαχείριση Σφαλμάτων στη VBA

Παράδειγμα χρήσης *OnError, Resume ,Err, Error(), Debug.Print* (2/5)



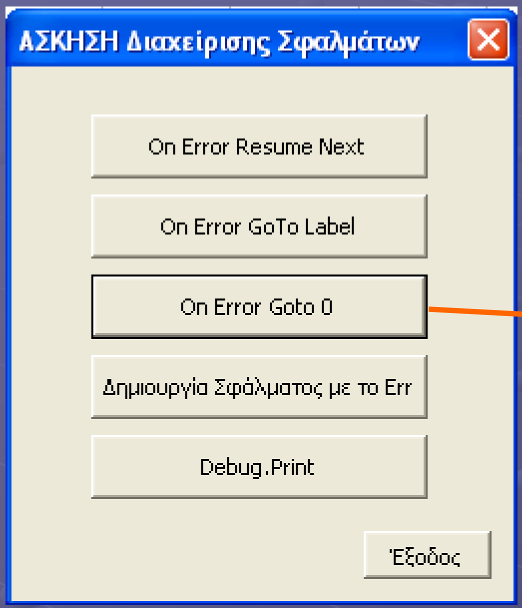

```
Private Sub GotoLabelButton_Click()
    Dim j(10) As Integer, Msg As String

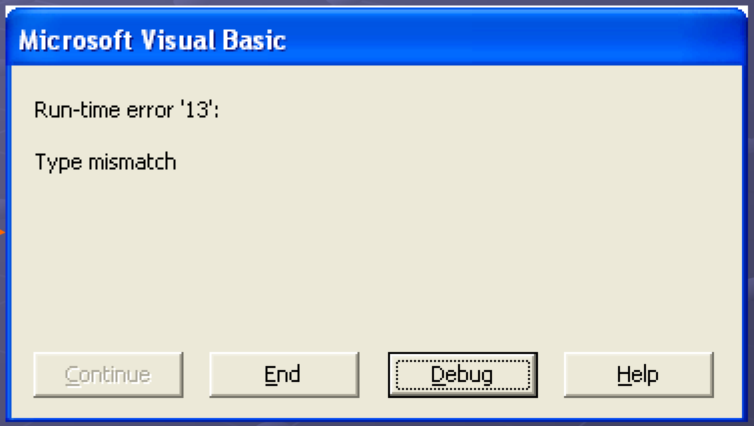
    On Error GoTo MyErrorHandler
    j(11) = 100    ' Προσπάθεια αποθήκευσης εκτός του Array

MyErrorHandler:
    Msg = "Ο Αριθμός του τελευταίου Σφάλαματος (Error) είναι " & Err.Number & _
      ". Το σχετικό μήνυμα Σφάλματος είναι: " & Err.Description  ' Χρήση του αντικειμένου Err
    MsgBox Msg, , "Προσοχή Σφάλμα"
End Sub
```

Διαχείριση Σφαλμάτων στη VBA

Παράδειγμα χρήσης *OnError, Resume, Err, Error(), Debug.Print* (3/5)





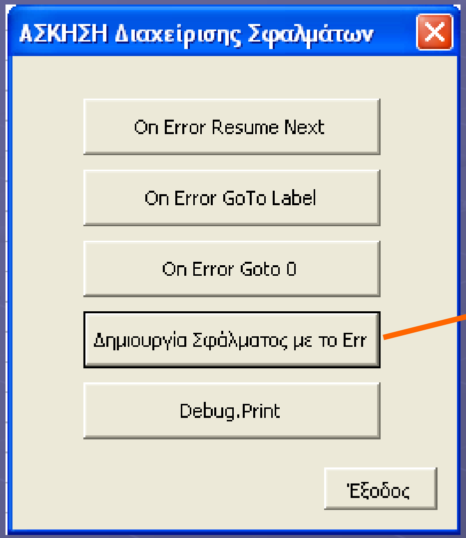

```
Private Sub GotoZeroButton_Click()
    Dim i As Integer, Msg As String

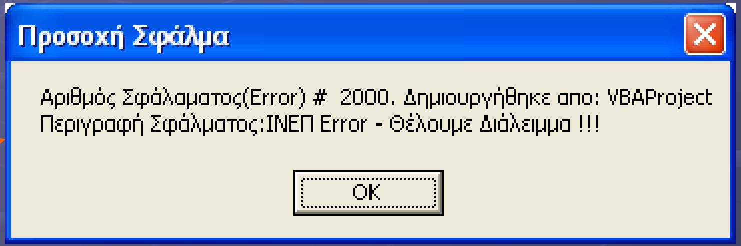
    On Error GoTo 0
    i = "String Test"    ' Προσπάθεια αποθήκευσης String σε Ακέραια Μεταβλητή

    Msg = "Ο Αριθμός του τελευταίου Σφάλματος (Error) είναι " & Err.Number & _
      ". Το σχετικό μήνυμα Σφάλματος είναι: " & Error(Err.Number)  ' Χρήση της συνάρτησης Error()
    MsgBox Msg, , "Προσοχή Σφάλμα"
End Sub
```

Διαχείριση Σφαλμάτων στη VBA

Παράδειγμα χρήσης *OnError, Resume ,Err, Error(), Debug.Print* (4/5)





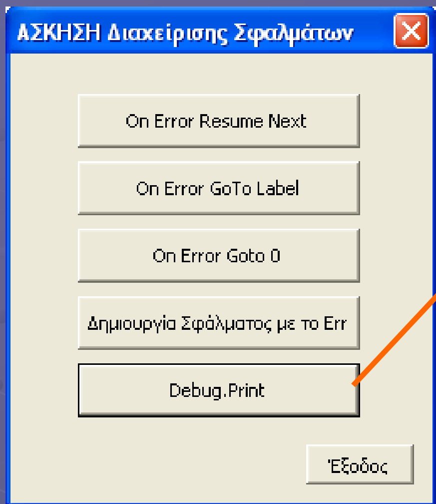

```
Private Sub ErrObjectButton_Click()
    Dim Msg As String

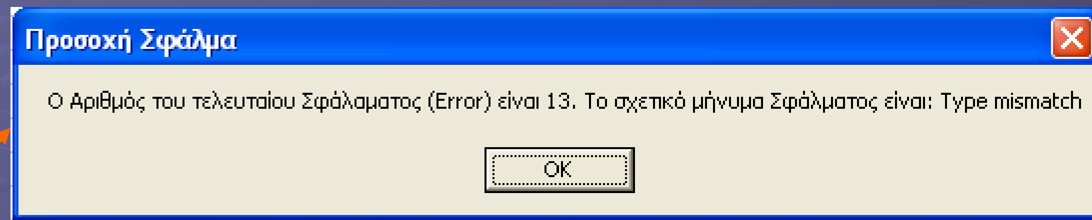
    On Error Resume Next
    Err.Clear                           ' Αρχικοποίηση του αντικειμένου Err
    Err.Raise 2000                      ' Δημιουργία "Τεχνητού" Σφάλματος
                                        ' Καθορισμός της Περιγραφής του "Τεχνητού" Σφάλματος
    Err.Description = "ΙΝΕΠ Error - Θέλουμε Διάλειμμα !!!"

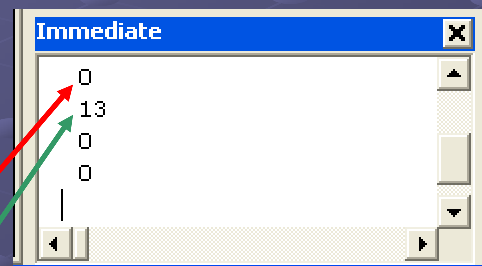
    If Err.Number <> 0 Then             ' Έλεγχος για ύπαρξη Σφάλματος
        Msg = "Αριθμός Σφάλαματος(Error) # " & Str(Err.Number) & ". Δημιουργήθηκε απο: " _
              & Err.Source & Chr(13) & "Περιγραφή Σφάλματος:" & Err.Description
        MsgBox Msg, , "Προσοχή Σφάλμα"
    End If
End Sub
```

Διαχείριση Σφαλμάτων στη VBA

Παράδειγμα χρήσης *OnError, Resume ,Err, Error(), Debug.Print* (5/5)








```
Private Sub CommandButton1_Click()
    Dim i As Integer, Msg As String

    On Error GoTo MyErrorHandler
    i = "String Test"          ' Το Σφάλμα συμβαίνει εδώ, αρα μεταβαίνουμε στο MyErrorHandler:
    Debug.Print i              ' Η εντολή εκτελείται μετά το Resume Next
    Debug.Print Err.Number     ' Η εντολή εκτελείται μετά το Resume Next, αρα το Err αρχικοποιήθηκε
    Exit Sub

MyErrorHandler:
    Msg = "Ο Αριθμός του τελευταίου Σφάλαματος (Error) είναι " & Err.Number & _
      ". Το σχετικό μήνυμα Σφάλματος είναι: " & Err.Description    ' Τα στοιχεία το Σφάλματος
    MsgBox Msg, , "Προσοχή Σφάλμα"
    Debug.Print i
    Debug.Print Err.Number     ' Θα εκτυπωθεί ο Αριθμός Σφάλματος στο παράθυρο "Immediate"
    Resume Next                ' Λόγω του "Resume Next" το Err θα αρχικοποιηθεί
End Sub
```

Σύνδεση με Βάσεις Δεδομένων (ΒΔ)
Τι είναι οι ΒΔ – Γιατί σύνδεση με ΒΔ

Μια ***Βάση Δεδομένων*** (Data Base) είναι ένα σύνολο δεδομένων αποθηκευμένων με συστηματικό τρόπο έτσι ώστε να είναι δυνατή η εύκολη, ασφαλής και αποτελεσματική διαχείριση και επεξεργασία τους.

Αν και υπάρχουν αρκετά μοντέλα Β.Δ. το πλέον διαδεδομένο είναι οι ***Σχεσιακές*** (Relational) Βάσεις Δεδομένων.

Το λογισμικό που χρησιμοποιείται για την διαχείριση μιας Σχεσιακής Βάσης Δεδομένων ονομάζεται ***RDBMS*** (Relational DB Management System). Γνωστά RDBMS είναι η Oracle, ο SQL Server, η MySQL, η MS-Access, κ.α.

Ειδικά για τις Σχεσιακές Βάσεις Δεδομένων, η *de facto standard* γλώσσα εντολών είναι η *Δομημένη Γλώσσα Ερωταπαντήσεων* ***SQL*** *(Structured Query Language).*

Οι Βάσεις Δεδομένων έχουν πολύ σημαντικά πλεονεκτήματα σχετικά με την αποθήκευση, ανάκτηση και διαχείριση των δεδομένων και γι αυτό σε όλα σχεδόν τα σύγχρονα προγράμματα είναι γραμμένα έτσι ώστε να μπορούν να λειτουργήσουν με οποιαδήποτε RDBMS. Αυτό επιτυγχάνεται με την χρήση ειδικών ***Προγραμματιστικών Διεπαφών*** (Programming Interfaces) και της γλώσσας SQL για την επικοινωνία με το εκάστοτε RDBMS.

Σύνδεση με Βάσεις Δεδομένων

Εισαγωγή στην SQL (1/4)

Η πιο γνωστή και ευρέως διαδεδομένη γλώσσα εντολών για τις σύγχρονες βάσεις δεδομένων είναι η *Δομημένη Γλώσσα Ερωταπαντήσεων* ***SQL*** *(**S**tructured **Q**uery **L**anguage)*, η οποία αποτελείται από τα εξής μέρη :

- ***DDL*** *(**D**ata **D**efinition **L**anguage, Γλώσσα Ορισμού Δεδομένων)*, με την οποία καθορίζουμε τις δομές και τα τμήματα μιας Βάσης Δεδομένων.
- ***DML*** *(**D**ata **M**anipulation **L**anguage, Γλώσσα Χειρισμού Δεδομένων)*, με την οποία επεξεργαζόμαστε τα δεδομένα μιας Βάσης Δεδομένων.
- ***DCL*** *(**D**ata **C**ontrol **L**anguage, Γλώσσα Ελέγχου Δεδομένων)*, με την οποία εξασφαλίζουμε την ασφάλεια και την ακεραιότητα των δεδομένων μιας Βάσης Δεδομένων.

Όλες οι σύγχρονες ΒΔ υποστηρίζουν την *SQL* και επομένως αν χρησιμοποιούμε την *SQL*, ο κώδικά που αναπτύσσουμε είναι μεταφέρσιμος και ανεξάρτητος από την εκάστοτε χρησιμοποιούμενη Βάση Δεδομένων.

Η ***SQL*** υποστηρίζει όλους τους γνωστούς *Τύπους Δεδομένων* (char(n), varchar(n), Int,, smallint, Numeric(p, d), real, double precision, float(n), date, time) αλλά μας επιτρέπει να ορίσουμε και δικούς μας τύπους, π.χ.

```
Create domain Surname char(20); ‘Ο τύπος Surname περιέχει 20 χαρακτ.
```

Σύνδεση με Βάσεις Δεδομένων
Εισαγωγή στην SQL (2/4)

Διαχείριση Πινάκων και Σχέσεων

```
                   ‘ Δημιουργία του Πίνακα Members
Create Table Members( MemberCode   integer not null,
                      Surname      char(20),
                      Name         char(20),
                      City         char(15),
                      BirthDate    date,
                      ClubCode     integer
                      primary key MemberCode )

                   ‘ Προσθήκη του Πεδίου AFM, μήκους 9 χαρακτήρων, στον Πίνακα Members
Alter table Members add (AFM char(9));

                   ‘ Διαγραφή του Πεδίου AFM από τον Πίνακα Members
Alter table Members drop AFM;

                   ‘ Διαγραφή του Πίνακα Members
Drop table Members;
```

Σύνδεση με Βάσεις Δεδομένων
Εισαγωγή στην SQL (3/4)

Εισαγωγή / Ενημέρωση / Διαγραφή δεδομένων

```
                                        ‘ Εισαγωγή στον Πίνακα Members μιας νέας εγγραφής
Insert into Members (MemberCode, Surname, Name, City, BirthDate, ClubCode )
     values  (126, ‘Καλπάκας’, ‘Κωνσταντίνος’, ‘ΑΘΗΝΑ’, null, 140);

                              ‘ Τροποποίηση της τιμής του Πεδίου ClubCode του Πίνακα Members
Update Members Set ClubCode = 150 Where ClubCode = 140;

                              ‘ Διαγραφή από τον Πίνακα Members όλων των μελών από ΑΘΗΝΑ
Delete from Members Where City = ‘ΑΘΗΝΑ’;
```

Σύνδεση με Βάσεις Δεδομένων
Εισαγωγή στην SQL (4/4)

Επιλογή Δεδομένων βάσει κριτηρίων

Η πιο σημαντική εντολή του DML μέρους της SQL είναι η εντολή Select:

Τρόπος σύνταξης:

Select *Πεδίο1, Πεδίο2, … Πεδίοn ή **
From *Πίνακας1, Πίνακας2, …, Πίνακαςm*
Where *Συνθήκη;*

Ο χαρακτήρας (*) είναι «μπαλαντέρ» και σημαίνει «όλα».

Για Παράδειγμα:

```
' Επιλογή των Πεδίων Name, Surname, City από τον Πίνακα Members
Select Name, Surname, City from Members;

' Επιλογή όλων των Πεδίων από τον Πίνακα Members
Select * from Members;

' Επιλογή όλων των Πεδίων από τον Πίνακα Members από ΑΘΗΝΑ
Select * from Members where City = 'ΑΘΗΝΑ';

' Επιλογή όλων των Πεδίων από τον Πίνακα Members από ΑΘΗΝΑ ή ΣΕΡΡΕΣ
Select * from Members where City = 'ΑΘΗΝΑ' OR City = 'ΣΕΡΡΕΣ' ;
```

Σύνδεση με Βάσεις Δεδομένων
Τρόποι και Τεχνολογίες Σύνδεσης με ΒΔ

Προκειμένου να κάνει δυνατό έναν εύκολο, αποτελεσματικό και ομοιόμορφο τρόπος πρόσβασης σε οποιοδήποτε πηγή δεδομένων, η Microsoft ανέπτυξε μια σειρά από αλληλοσυνδεδεμένες Τεχνολογίες, τις λεγόμενες ***MDAC*** (Microsoft Data Access Components). Αυτό περιλαμβάνει τα:

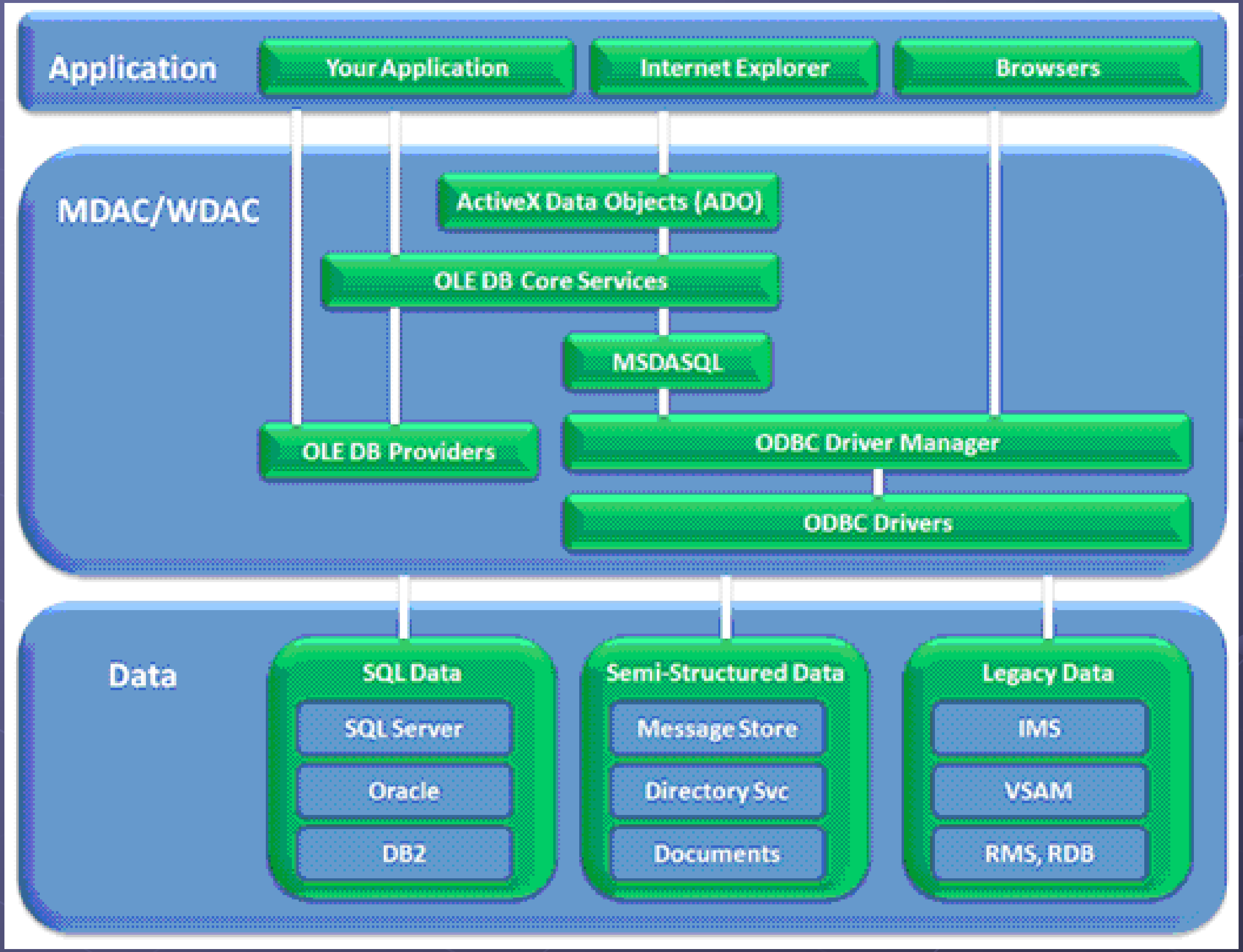
- ***ADO*** (ActiveX Data Objects),
- ***OLE DB*** και
- ***ODBC*** (Open Database Connectivity)

Το ***ADO*** είναι ένα προγραμματιστικό μοντέλο Υψηλού Επιπέδου, που χρησιμοποιεί το OLE DB. Είναι εύχρηστο και υποστηρίζει Αντικείμενα.

Το ***OLE DB*** είναι μια περιεκτική συλλογή από Διεπαφές COM που επιτρέπουν την πρόσβαση σε ένα ευρύ φάσμα πηγών Δεδομένων (Β.Δ., File System, κλπ).

Το ***ODBC*** είναι μια χαμηλού επιπέδου, standard, προγραμματιστική Διεπαφή που επιτρέπει την πρόσβαση σε Βάσεις Δεδομένων.

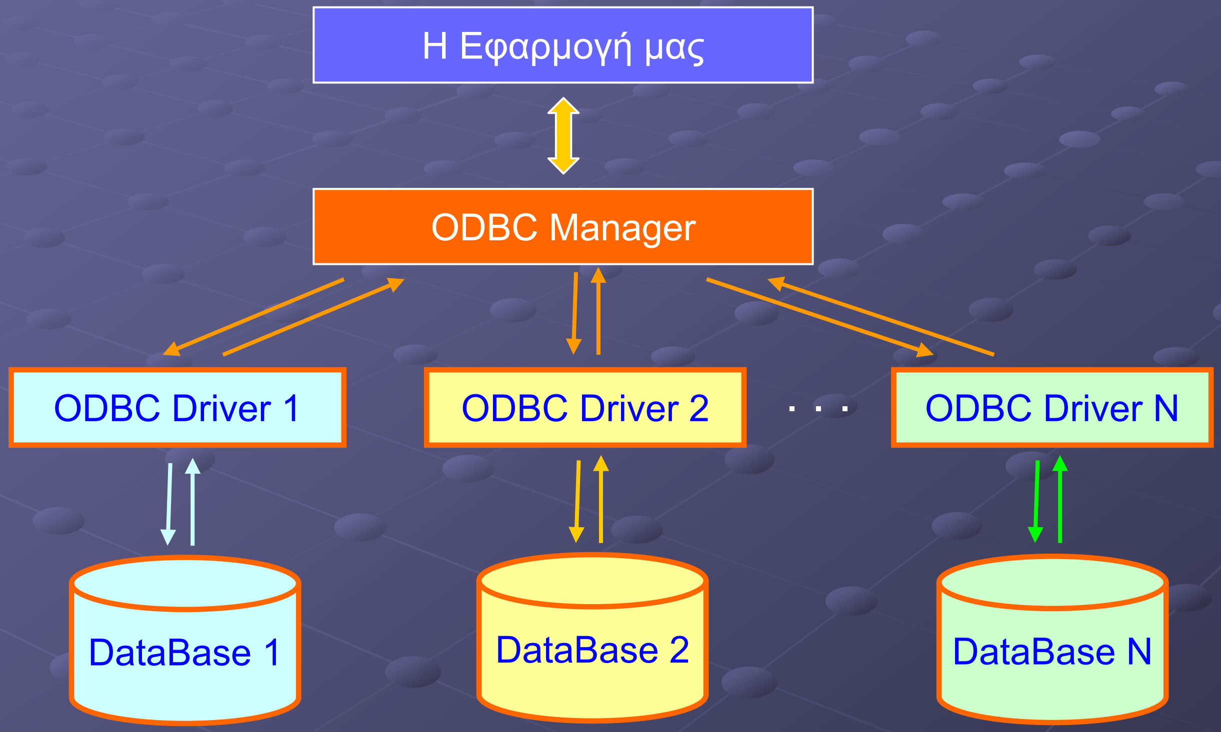
Σύνδεση με Βάσεις Δεδομένων



Σχηματική αναπαράσταση των Τεχνολογιών του MDAC και της αλληλεπίδρασής τους.

Σύνδεση με Βάσεις Δεδομένων - ODBC

Το μοντέλο **ODBC** (**O**pen **D**ata**B**ase **C**onnectivity) παρέχει μια standard διεπαφή για πρόσβαση σε Βάσεις Δεδομένων (*DataBases*). Δημιουργήθηκε από την Microsoft το 1992, ενώ η τελευταία έκδοσή του είναι η 3.8 (το 2009).

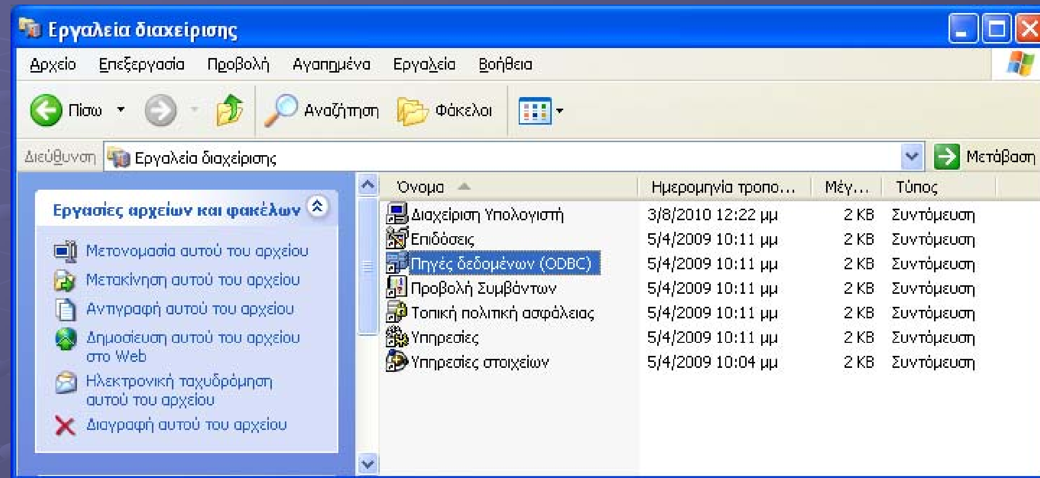


Στο μοντέλο ODBC η Εφαρμογή δεν «*επικοινωνεί*» απευθείας με την Β.Δ. αλλά πάντα μέσω του «*ODBC Manager*». Αυτός με την σειρά του «*επικοινωνεί*» με τον αντίστοιχο «*ODBC Driver*» της Β.Δ. για την οποία έχει κατασκευαστεί και της οποίας γνωρίζει την «γλώσσα». Έτσι η Εφαρμογή μας, είναι ανεξάρτητη από την Β.Δ. που χρησιμοποιείται.

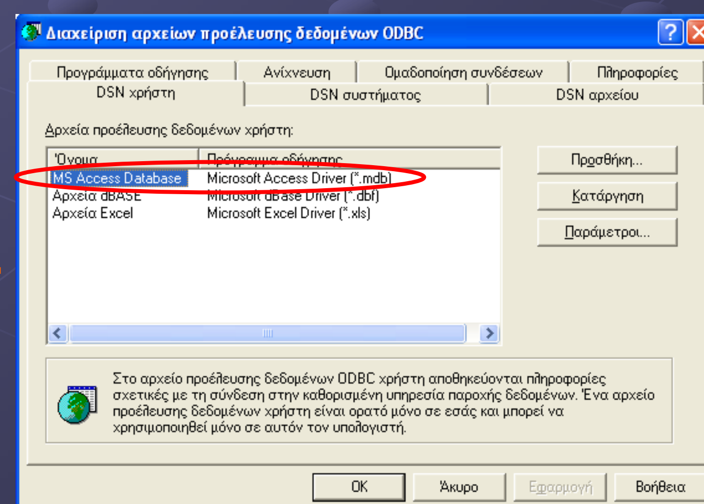
Σύνδεση με Βάσεις Δεδομένων – ODBC με DSN

Παράδειγμα Εισαγωγής Δεδομένων από Access (1/3)

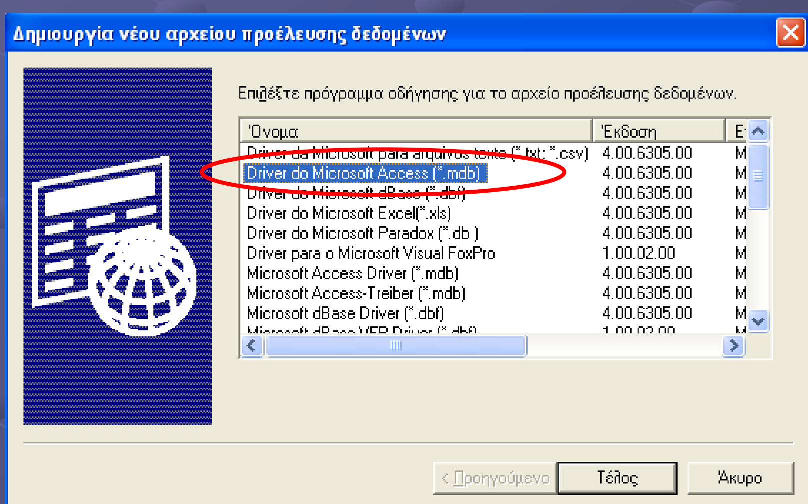
Για σύνδεση μέσω ODBC θα δημιουργήσουμε πρώτα ένα *DSN* (Data Source Name) για την Βάση μας. Αυτό γίνεται από τον *Πίνακα Ελέγχου* → *Εργαλεία Διαχείρισης* ακολουθώντας τα 6 που περιγράφονται σε αυτή και την επόμενη διαφάνεια.



(1)



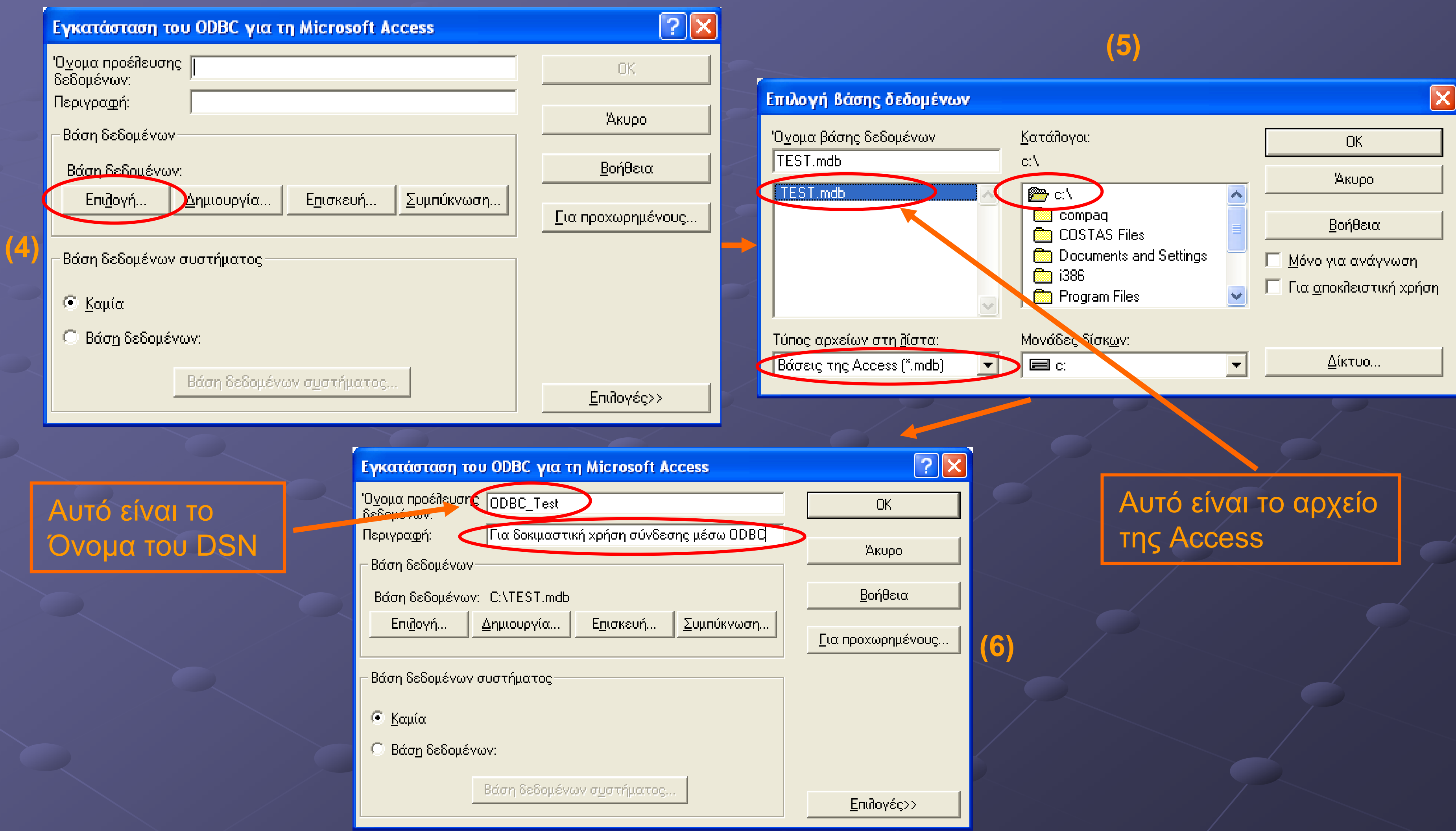
(2)



(3)

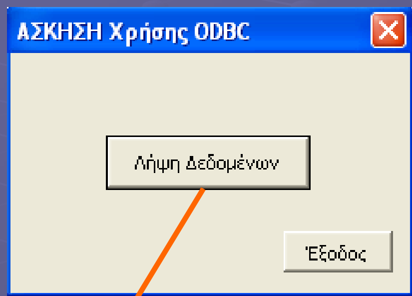
Σύνδεση με Βάσεις Δεδομένων – ODBC με DSN

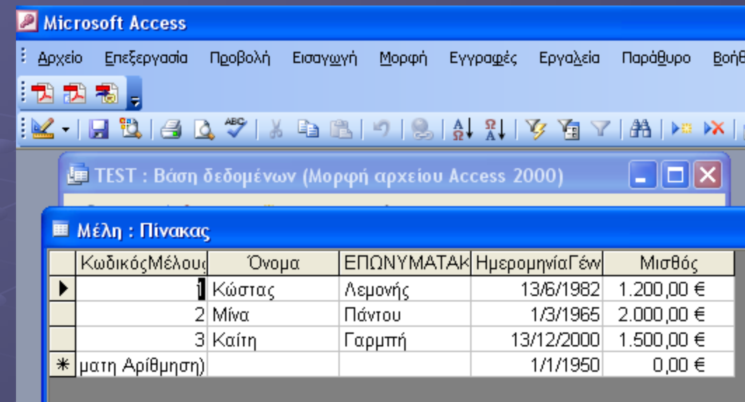
Παράδειγμα Εισαγωγής Δεδομένων από Access (2/3)



Σύνδεση με Βάσεις Δεδομένων – ODBC με DSN

Παράδειγμα Εισαγωγής Δεδομένων από Access (3/3)





Ο Πίνακας της Access

```
' Εισαγωγή δεδομένων από Πίνακα της Access στο Ενεργό Φύλλο μέσω ODBC & DSN
' ΠΡΟΣΟΧΗ: Πρέπει να έχει δημιουργηθεί απο το Control Panel ένα DSN με όνομα "ODBC_Test"
' ΠΡΟΣΟΧΗ: Η Φόρμα ΔΕΝ πρέπει να είναι Modal (Δηλ.: Showmodal = False)

Private Sub GetDataButton_Click()
    Dim DSNName, TblName, ConnString, SqlString As String
    Dim TargetRange As Range

    DSNName = "ODBC_Test"      ' Το Όνομα του DSN
    TblName = "Μέλη"           ' Το Όνομα του Πίνακα
    Set TargetRange = Range("B2") ' Σημείο αντιγραφής data

    SqlString = "SELECT * FROM " & TblName 'Εντολή SQL
    ConnString = "ODBC;DSN=" & DSNName      ' Εντολή σύνδεσης μέσω DSN

    With ActiveSheet.QueryTables.Add(Connection:=ConnString, _
                                Destination:=TargetRange, Sql:=SqlString)
        .Refresh
    End With
End Sub
```

F	G	H	I	J
ΚωδικόςΜέλους	Όνομα	Επώνυμο	ΗμερομηνίαΓέννησης	Μισθός
1	Κώστας	Λεμονής	13/6/1982 0:00	1200
2	Μίνα	Πάντου	1/3/1965 0:00	2000
3	Καίτη	Γαρμπή	13/12/2000 0:00	1500

Το ActiveSheet του Excel

Σύνδεση με Βάσεις Δεδομένων - ADO

Το μοντέλο *ADO* (**A**ctiveX **D**ata **O**bjects **)** είναι μια «*Προγραμματιστική Διεπαφή*» (*Programming Interface*) για την πρόσβαση των δεδομένων μίας Βάσης Δεδομένων, χωρίς να απαιτείται γνώση του πως έχει υλοποιηθεί η Β.Δ.

Δημιουργήθηκε από την Microsoft το 1996 και είναι ένα Microsoft «*ActiveX component*», το οποίο εγκαθίσταται αυτόματα με τον "*IIS*" της Microsoft αλλά είναι και άμεσα διαθέσιμο στη VBA του Excel αρκεί να προσθέσουμε μια σχετική «*Αναφορά*» (*reference*) στην «*Βιβλιοθήκη Αντικειμένων ADO*» (ADO Object Library). Αυτό γίνεται ως εξής:

> Από το μενού «*Tools*» του περιβάλλοντος VBA επιλέγουμε «*References…*» και επιλέγουμε το «*Microsoft ActiveX Data Objects 2.8 Library*» (ή νεότερη)

Το *ADO* είναι ένα «*COM component*» βασισμένο στο *ODBC*. Παρέχει την λειτουργικότητα του ODBC αλλά με αντικειμενοστραφή τρόπο. Είναι αρκετά ευκολότερο στην χρήση του, ενώ υστερεί ελάχιστα στην αποδοτικότητα.

Το ADO είναι απόγονος των καταργηθέντων RDO (Remote Data Objects) και DAO (Data Access Objects).

Σύνδεση με Βάσεις Δεδομένων - ADO

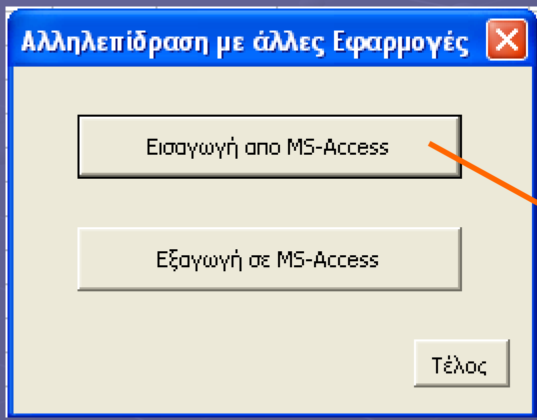
Τα βήματα που ακολουθούμε στο μοντέλο *ADO* για την *Ανάκτηση Δεδομένων* από μία Βάση Δεδομένων είναι τα εξής *:

1. Δημιουργία *Αντικειμένου Σύνδεσης* (Connection object)
   (Σύνταξη: Set MyConn = New ADODB.Connection)
2. Δημιουργία *Αντικειμένου Recordset* (Recordset object)
   (Σύνταξη: Set MyRecSet = New ADODB.Recordset)
3. Άνοιγμα της *Σύνδεσης*
   (Σύνταξη: MyConn .Open ConnectionString, UserID, Password, Options)
4. Άνοιγμα / *Γέμισμα* του *Recordset*
   (Σύνταξη: MyRecSet.Open Source, Connection, CursorType, LockType, Options)
5. *Αντιγραφή των περιεχομένων* του *Recordset* στο Excel
   (Σύνταξη: RangeExpression.CopyFromRecordset MyRecSet)
6. Κλείσιμο του *Recordset* *(Σύνταξη: MyRecSet.Close)*
7. Κλείσιμο της *Σύνδεσης* *(Σύνταξη: MyConn.Close)*
8. Καταστροφή (*free*) του *Αντικειμένου Recordset* *(Σύνταξη: Set MyRecSet = Nothing)*
9. Καταστροφή (*free*) του *Αντικειμένου Σύνδεσης* *(Σύνταξη: Set MyConn = Nothing)*

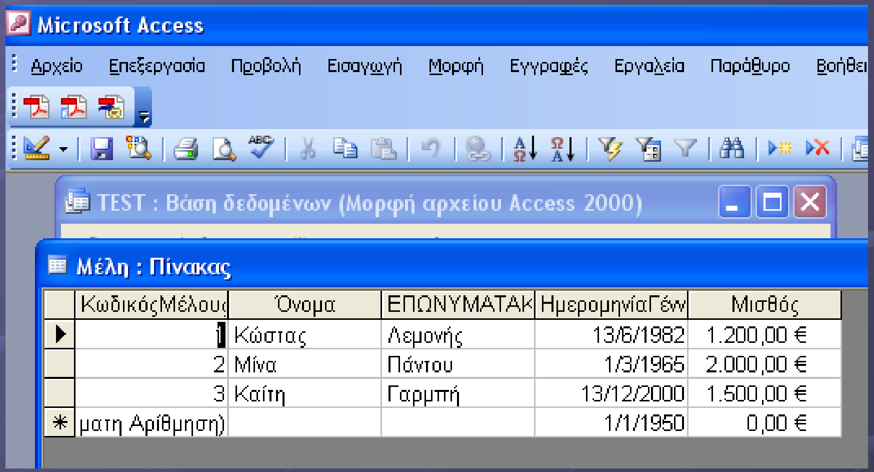
* Υπάρχουν πολλοί εναλλακτικοί τρόποι στο μοντέλο ADO, αλλά ο ανωτέρω είναι ίσως ο απλούστερος

Σύνδεση με Βάσεις Δεδομένων – ADO

Παράδειγμα Εισαγωγής Δεδομένων από Access (1/2)



(1)



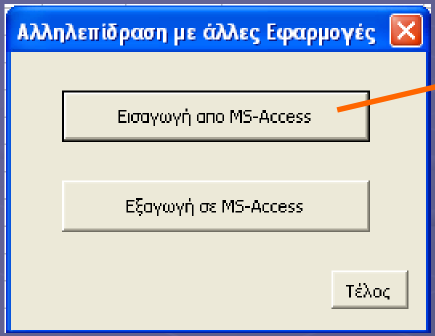
Κάνοντας κλικ στο κουμπί «*Εισαγωγή από MS-Access*» τα δεδομένα από τον Πίνακα «*Μέλη*» της Βάσης Δεδομένων «*Temp*» θα αντιγραφούν στο Ενεργό Φύλλο (ActiveSheet) του Excel.

(2)

Εισαγωγή δεδομένων στο Excel από Β.Δ. της MS Access

ΚωδικόςΜέλους	Όνομα	Επώνυμο	ΗμερομηνίαΓέννησης	Μισθός
1	Κώστας	Λεμονής	13/6/1982	1200
2	Μίνα	Πάντου	1/3/1965	2000
3	Καίτη	Γαρμπή	13/12/2000	1500

Σύνδεση με Βάσεις Δεδομένων – ADO
Παράδειγμα Εισαγωγής Δεδομένων από Access (2/2)




```
Private Sub cmdImportFromAccess_Click()   ' Κουμπί "Εισαγωγή από MS-Access"
    ADOImportFromAccessTable "C:\Test.mdb", "Μέλη", Range("F10")
End Sub
```

```
Sub ADOImportFromAccessTable(DBFullName As String, TableName As String, TargetRange As Range)
    Dim Conn As ADODB.Connection, RecSet As ADODB.Recordset, intColIndex As Integer

    Set Conn = New ADODB.Connection      ' Δημιουργία του Αντικειμένου Σύνδεσης
    Set RecSet = New ADODB.Recordset     ' Δημιουργία του Αντικειμένου Recordset
                                         ' Άνοιγμα της Σύνδεσης με την Βάση Δεδομένων
    Conn.Open "Provider=Microsoft.Jet.OLEDB.4.0; Data Source=" & DBFullName & ";"

    With RecSet
                                         ' Άνοιγμα του Πίνακα και επιλογή όλων των Εγγραφών
        .Open "SELECT * FROM " & TableName, Conn, , , adCmdText

        For intColIndex = 0 To RecSet.Fields.Count - 1    ' Εισαγωγή των Ονομάτων Πεδίων
            TargetRange.Offset(0, intColIndex).Value = RecSet.Fields(intColIndex).Name
        Next
                                         ' Επικόλληση των δεδομένων του Recordset
        TargetRange.Offset(1, 0).CopyFromRecordset RecSet
    End With

    ActiveSheet.UsedRange.EntireColumn.AutoFit      'Προσαρμογή του πλάτους Στηλών

    RecSet.Close                         ' Κλείσιμο του Αντικειμένου Recordset
    Set RecSet = Nothing                 ' Απελευθέρωση του Αντικειμένου Recordset
    Conn.Close                           ' Κλείσιμο του Αντικειμένου Σύνδεσης
    Set Conn = Nothing                   ' Απελευθέρωση του Αντικειμένου Σύνδεσης
End Sub
```

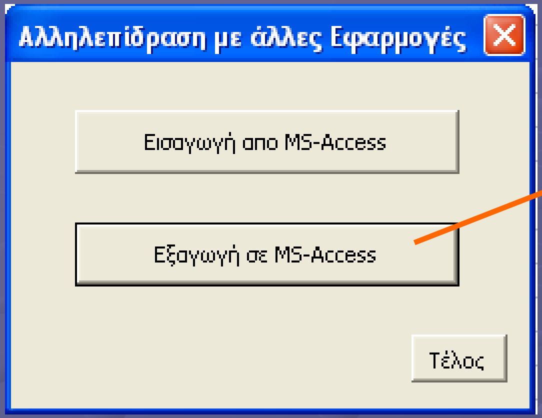
Σύνδεση με Βάσεις Δεδομένων - ADO

Τα βήματα που ακολουθούμε στο μοντέλο *ADO* για την *Εγγραφή Δεδομένων* σε μία Βάση Δεδομένων είναι τα εξής:

1. Δημιουργία *Αντικειμένου Σύνδεσης* (Connection object)
   (Σύνταξη: Set MyConn = New ADODB.Connection)
2. Δημιουργία *Αντικειμένου Recordset* (Recordset object)
   (Σύνταξη: Set MyRecSet = New ADODB.Recordset)
3. Άνοιγμα της *Σύνδεσης*
   (Σύνταξη: MyConn .Open ConnectionString, UserID, Password, Options)
4. Άνοιγμα του *Recordset* και του *Πίνακα* της Β.Δ.
   (Σύνταξη: MyRecSet.Open TableName, Connection, CursorType, LockType)
5. *Δημιουργία Νέας Εγγραφής* στο *Recordset* *(Σύνταξη: MyRecSet .AddNew)*
6. *Καταχώρηση Τιμών* στα Πεδία της Νέας Εγγραφής του *Recordset*
   (Σύνταξη: MyRecSet .Fields(“Επώνυμο”) = Range(“A5”).Value
7. *Ενημέρωση* της Β.Δ. από το *Recordset* *(Σύνταξη: MyRecSet .Update)*
8. Κλείσιμο του *Recordset* *(Σύνταξη: MyRecSet.Close)*
9. Κλείσιμο της *Σύνδεσης* *(Σύνταξη: MyConn.Close)*
10. Καταστροφή (*free*) του *Αντικειμένου Recordset* *(Σύνταξη: Set MyRecSet = Nothing)*
11. Καταστροφή (*free*) του *Αντικειμένου Σύνδεσης* *(Σύνταξη: Set MyConn = Nothing)*

Σύνδεση με Βάσεις Δεδομένων – ADO

Παράδειγμα εξαγωγής Δεδομένων σε Access (1/2)



(1)

	A	B	C	D	E
1					
2	ΚωδικόςΜέλους	Όνομα	Επώνυμο	ΗμερομηνίαΓέννησης	Μισθός
3	4	Λινα	Πουλή	18/5/1963	1600
4	5	Μαρία	Θάνου	20/8/1953	2600
5	6	Πόπη	Καπνιάρη	9/9/1984	3600
6					

(2)



TEST : Βάση δεδομένων (Μορφή αρχείου Access 2000)

Μέλη : Πίνακας

ΚωδικόςΜέλους	Όνομα	ΕΠΩΝΥΜΑΤΑΚ	ΗμερομηνίαΓέν	Μισθός
1	Κώστας	Λεμονής	13/6/1982	1.200,00 €
2	Μίνα	Πάντου	1/3/1965	2.000,00 €
3	Καίτη	Γαρμπή	13/12/2000	1.500,00 €
ματη Αρίθμηση)			1/1/1950	0,00 €

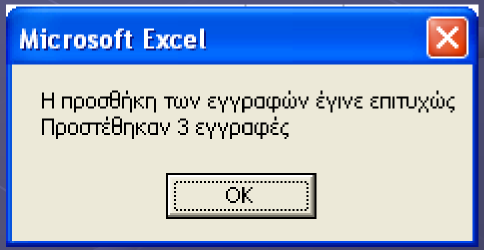
(3)

TEST : Βάση δεδομένων (Μορφή αρχείου Access 2000)

Μέλη : Πίνακας

ΚωδικόςΜέλους	Όνομα	ΕΠΩΝΥΜΑΤΑΚ	ΗμερομηνίαΓέν	Μισθός
1	Κώστας	Λεμονής	13/6/1982	1.200,00 €
2	Μίνα	Πάντου	1/3/1965	2.000,00 €
3	Καίτη	Γαρμπή	13/12/2000	1.500,00 €
4	Λινα	Πουλή	18/5/1963	1.600,00 €
5	Μαρία	Θάνου	20/8/1953	2.600,00 €
6	Πόπη	Καπνιάρη	9/9/1984	3.600,00 €
ματη Αρίθμηση)			1/1/1950	0,00 €

(4)



Κάνοντας κλικ στο κουμπί «*Εξαγωγή σε MS-Access*» τα δεδομένα από το ActiveSheet του Excel εισ'αγονται στον Πίνακα «*Μέλη*» της Βάσης Δεδομένων «*Temp*».

Σύνδεση με Βάσεις Δεδομένων – ADO

Παράδειγμα εξαγωγής Δεδομένων σε Access (2/2)




```
Sub ADOFromExcelToAccess(DBFullName As String, TblName As String)
    Dim Conn As ADODB.Connection, RecSet As Recordset
    Dim StartRow As Long, SuccessCnt As Integer

    Set Conn = New ADODB.Connection      ' Δημιουργία του Αντικειμένου Σύνδεσης
    Set RecSet = New ADODB.Recordset     ' Δημιουργία του Αντικειμένου Recordset
                                         ' Άνοιγμα της Σύνδεσης με την Βάση Δεδομένων
    Conn.Open "Provider=Microsoft.Jet.OLEDB.4.0; Data Source=" & DBFullName & ";"
                                         ' Άνοιγμα του Πίνακα της ΒΔ για προσθήκη εγγραφών
    RecSet.Open TblName, Conn, CursorType:=adOpenKeyset, LockType:=adLockOptimistic
    On Error GoTo ErrorHandle            ' Για την περίπτωση προβλήματος στην Ενημέρωση της ΒΔ
    StartRow = 3                         ' Η 1η σειρά του ActiveSheet απ΄όπου θα ληφθούν δεδομένα
    SuccessCnt = 0                       ' Μετρητής Επιτυχημένων εγγραφών στη Βάση
    Do While Len(Range("A" & StartRow).Formula) > 0      ' Επανάληψη έως ότου βρεθεί το πρώτο
                                                         ' κενό κελί στην στήλη Α
        With RecSet
            .AddNew                      ' Δημιουργία νέας Εγγραφής στο Recordset
                                         ' Προσθήκη Τιμών σε κάθε Πεδίο της Εγγραφής
            .Fields("ΚωδικόςΜέλους") = Range("A" & StartRow).Value 'PRIMARY KEY - NO DUBLICATES !!!
            .Fields("Όνομα") = Range("B" & StartRow).Value
            .Fields("Επώνυμο") = Range("C" & StartRow).Value
            .Fields("ΗμερομηνίαΓέννησης") = Range("D" & StartRow).Value
            .Fields("Μισθός") = Range("E" & StartRow).Value
            .Update                      ' Ενημέρωση της Β.Δ. απο το Recordset (με τη νέα εγγραφή)
        End With
        StartRow = StartRow + 1          ' Επόμενη Γραμμή
        SuccessCnt = SuccessCnt + 1      ' Αύξηση του Μετρητή Επιτυχών Εγγραφών
    Loop
    RecSet.Close                         ' Κλείσιμο του του Αντικειμένου Recordset
    Set RecSet = Nothing                 ' Απελευθέρωση του Αντικειμένου Recordset
    Conn.Close                           ' Κλείσιμο του Αντικειμένου Σύνδεσης
    Set Conn = Nothing                   ' Απελευθέρωση του Αντικειμένου Σύνδεσης
    GoTo TELOS
ErrorHandle:
    MsgBox "Πρόβλημα στην ενημέρωση της Β.Δ.!" & vbCrLf & "Προστέθηκαν " & SuccessCnt & " εγγραφές"
    Exit Sub
TELOS:
    MsgBox "Επιτυχής εγγραφή!" & vbCrLf & "Προστέθηκαν " & SuccessCnt & " εγγραφές"
End Sub
```

5^{η} Ημέρα Επιμόρφωσης

- Συνεργασία με άλλες Εφαρμογές του Office
- Μελέτη Περίπτωσης (Επαναληπτική Άσκηση)
- Αξιολόγηση Προγράμματος & Εκπαιδευομένων

Συνεργασία με άλλες Εφαρμογές
MS-Word, MS-PowerPoint, MS-Outlook

Μέσω της VBA αυτής μπορούμε να επιτύχουμε αμφίδρομη επικοινωνία μεταξύ του Excel και άλλων Εφαρμογών του MS-Office (π.χ. MS-Word, MS-PowerPoint, MS-Outlook κλπ).

Παρακάτω βλέπουμε τμήματα κώδικα VBA για την δημιουργία μέσω της VBA, εγγράφου Word και Παρουσίασης PowerPoint και την εισαγωγή κειμένου σ' αυτά.

Κλήση του MS-Word και δημιουργία Νέου Document:

```
Set WordApp = CreateObject("Word.Application")          ' Κλήση του Word
WordApp.Visible = True                                  ' Για να εμφανιστεί το WORD
Set WordDoc = WordApp.Documents.Add                     ' Δημιουργία Νέου Document
```

Δημιουργία Νέας Παραγράφου και Εισαγωγή κειμένου:

```
Set WordParagraph1 = WordDoc.Content.Paragraphs.Add     ' Δημιουργία Νέας Παραγράφου
WordParagraph1.Range.Text = "ΕΠΙΚΕΦΑΛΙΔΑ από Excel VBA" ' Εισαγωγή κειμένου
```

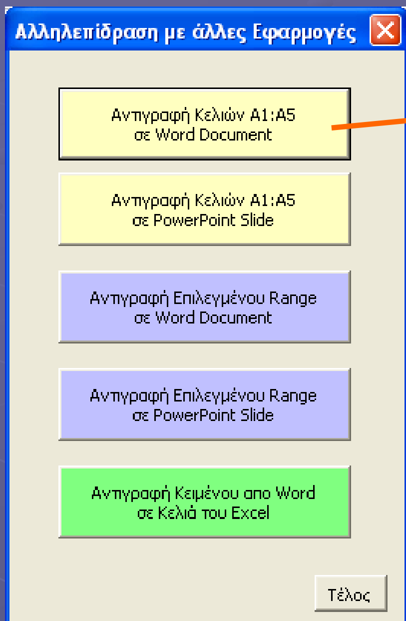
Κλήση του MS-PowerPoint και δημιουργία Νέας Παρουσίασης:

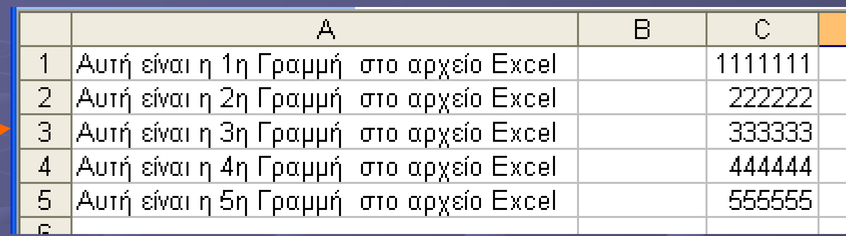
```
Set PPTapp = CreateObject("PowerPoint.Application") ' Κλήση του PowerPoint
PPTapp.Visible = True                               ' Για να εμφανιστεί το PowerPoint
Set PPTpresent = PPTapp.Presentations.Add           ' Δημιουργία Νέας Παρουσίασης
```

Δημιουργία Νέας Διαφάνειας (Slide) και Εισαγωγή κειμένου:

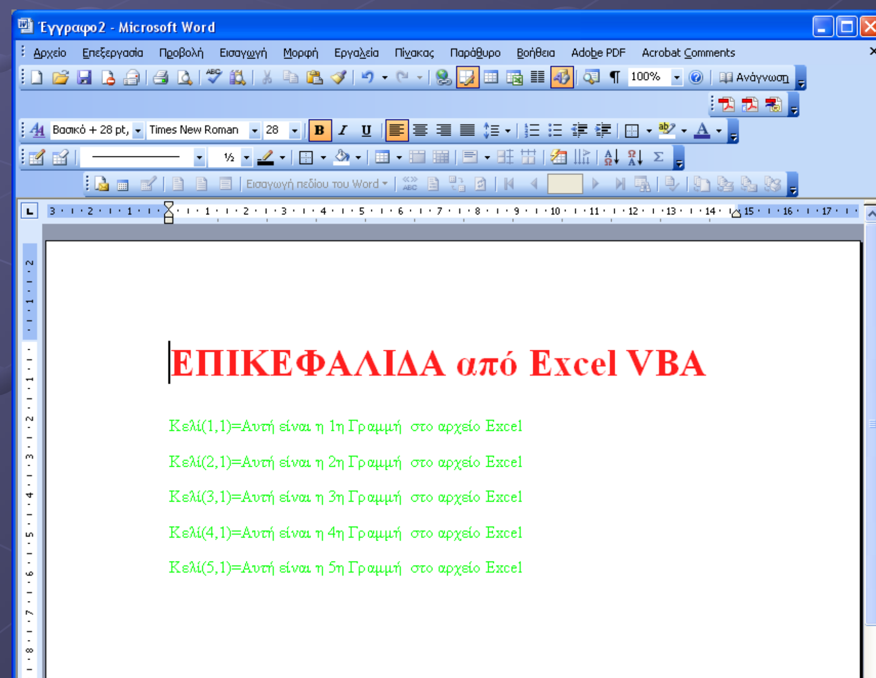
```
Set PPTslide = PPTpresent.Slides.Add(Index:=PPTpresent.Slides.Count + 1, _
                                     Layout:=ppLayoutText)      ' Προσθήκη Slide
PPTslide.Shapes.Title.TextFrame.TextRange = "ΕΠΙΚΕΦΑΛΙΔΑ Slide"  ' Κείμενο Slide
```

Συνεργασία με άλλες Εφαρμογές
Με MS-Word και MS-PowerPoint (Παράδειγμα) - (1/6)





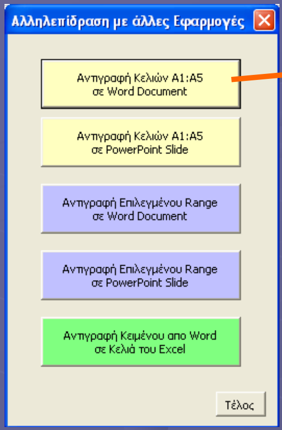
	A	B	C
1	Αυτή είναι η 1η Γραμμή στο αρχείο Excel		1111111
2	Αυτή είναι η 2η Γραμμή στο αρχείο Excel		222222
3	Αυτή είναι η 3η Γραμμή στο αρχείο Excel		333333
4	Αυτή είναι η 4η Γραμμή στο αρχείο Excel		444444
5	Αυτή είναι η 5η Γραμμή στο αρχείο Excel		555555



Στη Άσκηση αυτή θέλουμε το περιεχόμενο των κελιών A1 έως A5 να αντιγραφεί σε ένα Word Document που θα δημιουργηθεί.

Συνεργασία με άλλες Εφαρμογές

Με MS-Word και MS-PowerPoint (Παράδειγμα) - (2/6)




```
Private Sub cmdStartWord_Click()
    Call CopyCellsToWord
End Sub
```

```
' Αντιγραφή του περιεχομένου των Κελιών A1 έως A5 σε Word Document
Public Sub CopyCellsToWord()
    Dim WordApp, WordDoc, WordParagraph1, WordParagraph2 As Object

    Set WordApp = CreateObject("Word.Application")          ' Κλήση του Word
    WordApp.Visible = True                                   ' ΑΠΑΙΤΕΙΤΑΙ για να εμφανιστεί το WORD
    Set WordDoc = WordApp.Documents.Add                      ' Δημιουργία Νέου Document

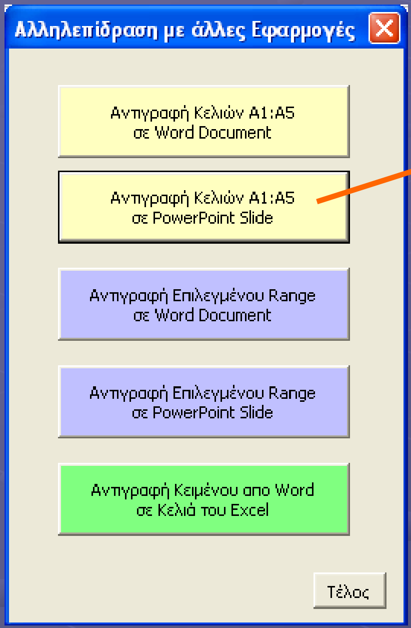
    'Εισαγωγή και Μορφοποίηση Επικεφαλίδας, στο Word Document.
    Set WordParagraph1 = WordDoc.Content.Paragraphs.Add      ' Δημιουργία Νέας Παραγράφου
    WordParagraph1.Range.Text = "ΕΠΙΚΕΦΑΛΙΔΑ από Excel VBA"
    WordParagraph1.Range.Font.Size = 28                      ' Font μεγέθους 28 στ.
    WordParagraph1.Range.Font.Bold = True                    ' Font Bold
    WordParagraph1.Range.Font.Color = RGB(255, 30, 30)       ' Χρώμα Font (απόχρωση του Κόκκινου)
    WordParagraph1.Format.SpaceAfter = 24                    ' 24 στ. κενό μετά την Παράγραφο.
    WordParagraph1.Range.InsertParagraphAfter                ' Εισαγωγή της Παραγράφου μετά την τρέχουσα

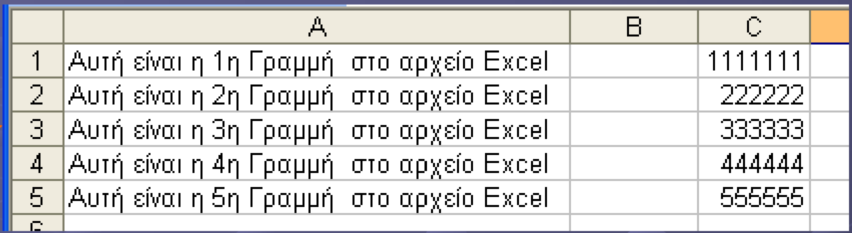
    'Εισαγωγή και Μορφοποίηση Κελιών A1 έως A5, στο Word Document
    For i = 1 To 5
        Set WordParagraph2 = WordDoc.Content.Paragraphs.Add ' Δημιουργία Νέας Παραγράφου
        WordParagraph2.Range.Text = "Κελί(" & i & ",1)=" & Cells(i, 1).Value
        WordParagraph1.Range.Font.Size = 12                  ' Font μεγέθους 12 στ.
        WordParagraph1.Range.Font.Bold = False               ' Font Κανονικό
        WordParagraph1.Range.Font.Color = RGB(0, 255, 0)     ' Χρώμα Font (Πράσινο)
        WordParagraph2.Format.SpaceAfter = 12                ' 12 στ. κενό μετά την Παράγραφο.
        WordParagraph2.Range.InsertParagraphAfter            ' Εισαγωγή της Παραγράφου μετά την τρέχουσα
    Next i

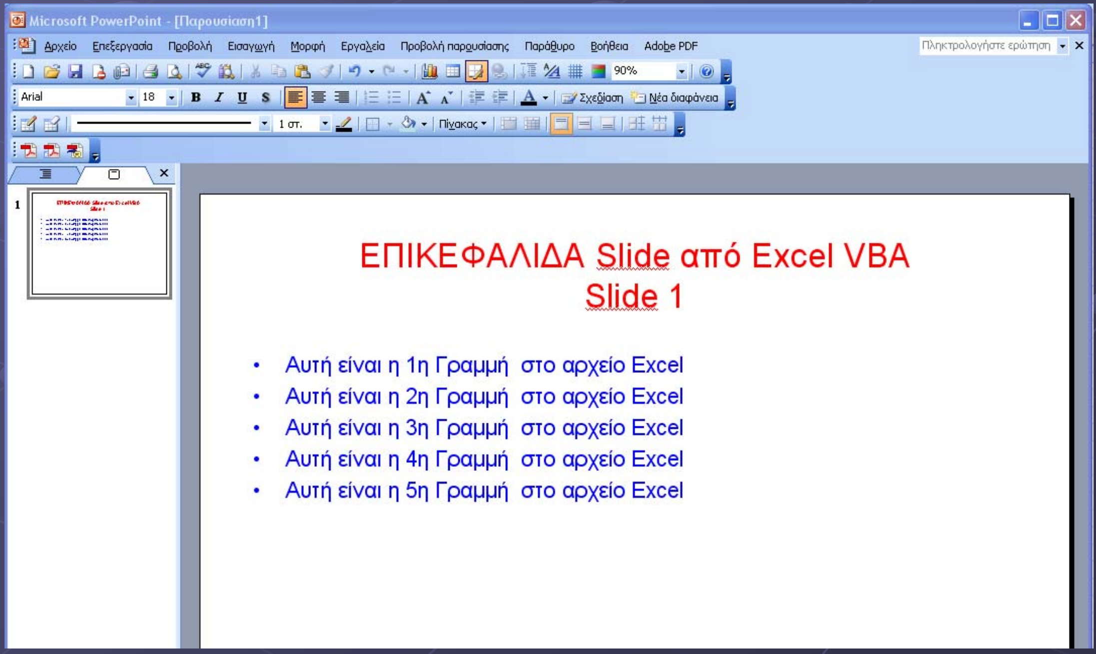
End Sub
```

Συνεργασία με άλλες Εφαρμογές

Με MS-Word και MS-PowerPoint (Παράδειγμα) - (3/6)



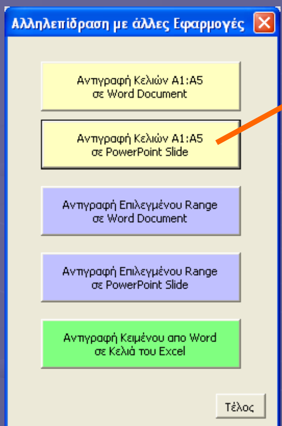




Στη Άσκηση αυτή θέλουμε το περιεχόμενο των κελιών A1 έως A5 να αντιγραφεί σε μία Διαφάνεια μιας νέας Παρουσίασης PowerPoint που θα δημιουργηθεί.

Συνεργασία με άλλες Εφαρμογές

Με MS-Word και MS-PowerPoint (Παράδειγμα) - (4/6)




```
Private Sub cmdStartPowerPoint_Click()
    Call CopyCellsToPowerPoint
End Sub
```

```
' Αντιγραφή του περιεχομένου των Κελιών A1 έως A5 σε PowerPoint Slide
Public Sub CopyCellsToPowerPoint()
    Dim PPTapp, PPTpresent, PPTslide As Object
    Dim SlideBullets As String

    Set PPTapp = CreateObject("PowerPoint.Application") ' Κλήση του PowerPoint
    PPTapp.Visible = True                               ' Για να εμφανιστεί το PoerPoint
    Set PPTpresent = PPTapp.Presentations.Add           ' Δημιουργία Νέας Παρουσίασης

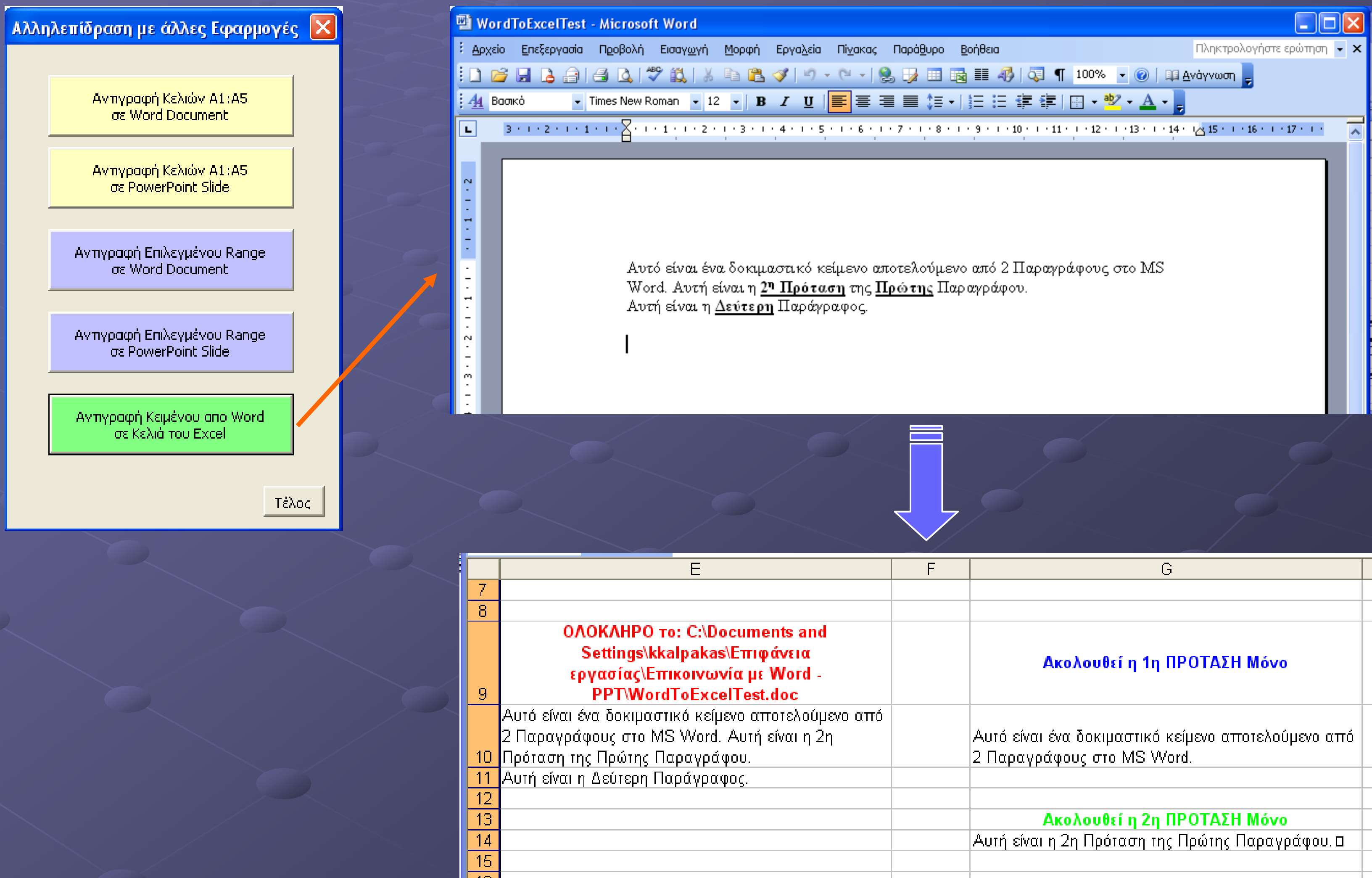
    'Εισαγωγή και Μορφοποίηση Επικεφαλίδας, στο PowerPoint Slide.
    Set PPTslide = PPTpresent.Slides.Add(Index:=PPTpresent.Slides.Count + 1, _
                                   Layout:=ppLayoutText)             ' Προσθήκη Slide
    PPTslide.Shapes.Title.TextFrame.TextRange = "ΕΠΙΚΕΦΑΛΙΔΑ Slide από Excel VBA" & vbNewLine & "Slide 1"
    PPTslide.Shapes.Title.TextFrame.TextRange.Font.Size = 28             ' Font μεγέθους 28 στ.
    PPTslide.Shapes.Title.TextFrame.TextRange.Font.Color = RGB(255, 0, 0)    ' Χρώμα Font (Κόκκινο)

    'Εισαγωγή και Μορφοποίηση Κελιών A1 έως A5, στο PowerPoint Slide
    SlideBullets = ""
    For i = 1 To 5                                                   ' Bullets
        SlideBullets = SlideBullets & Cells(i, 1) & vbNewLine
    Next i
    PPTslide.Shapes(2).TextFrame.TextRange = SlideBullets
    PPTslide.Shapes(2).TextFrame.TextRange.Font.Size = 18               ' Font μεγέθους 28 στ.
    PPTslide.Shapes(2).TextFrame.TextRange.Font.Color = RGB(0, 0, 255)      ' Χρώμα Font (Μπλέ)

End Sub
```

Συνεργασία με άλλες Εφαρμογές

Με MS-Word και MS-PowerPoint (Παράδειγμα) - (5/6)



Συνεργασία με άλλες Εφαρμογές

Με MS-Word και MS-PowerPoint - (Παράδειγμα) - (6/6)




```
Private Sub WordToExcelButton_Click()
    Call CopyFromWordToExcel
End Sub
```

```
' Αντιγραφή Κειμένου απο Word Document σε κελιά του Excel
Public Sub CopyFromWordToExcel()

    Dim MyWd As Object, WordFileName As String, FlName As String

    FlName = "WordToExcelTest.doc"                                 ' Όνομα αρχείου Word
    WordFileName = ThisWorkbook.Path & "\" & FlName                ' Πλήρες Όνομα αρχείου Word
    Set MyWd = GetObject(WordFileName)                             ' Άνοιγμα αρχείου Word

    ActiveSheet.Range("E9").Value = "ΟΛΟΚΛΗΡΟ το: " & WordFileName    ' Επικεφαλίδα στο κελί E9
    ActiveSheet.Range("E9").Font.Color = RGB(255, 0, 0)            ' Κόκκινο Χρώμα
    ActiveSheet.Range("E9").Font.Bold = True                       ' Bold
    ActiveSheet.Range("E9").VerticalAlignment = xlCenter           ' Κάθετη στοίχιση Κέντρο
    ActiveSheet.Range("E9").HorizontalAlignment = xlCenter         ' Οριζόντια στοίχιση Κέντρο
    MyWd.ActiveWindow.Selection.WholeStory                         ' Επιλογή Ολόκληρου του εγγράφου
    MyWd.ActiveWindow.Selection.Copy                               ' Αντριγραφή της επιλογής
    ActiveSheet.Range("E10").PasteSpecial xlPasteValues            ' Επικόλληση αρχίζοντας στο F10
    ActiveSheet.Range("E10").EntireColumn.ColumnWidth = 45         ' Αλλαγή Πλάτους Στήλης σε 45 στ.
    ActiveSheet.Range("E1:E100").WrapText = True                   ' Αναδίπλωση κειμένου

    ActiveSheet.Range("G9").Value = "Ακολουθεί η 1η ΠΡΟΤΑΣΗ Μόνο"   ' Επικεφαλίδα στο κελί G9
    ActiveSheet.Range("G9").Font.Color = RGB(0, 0, 255)            ' Μπλέ Χρώμα
    ActiveSheet.Range("G9").Font.Bold = True                       ' Bold
    ActiveSheet.Range("G9").VerticalAlignment = xlCenter           ' Κάθετη στοίχιση Κέντρο
    ActiveSheet.Range("G9").HorizontalAlignment = xlCenter         ' Οριζόντια στοίχιση Κέντρο
    MyWd.ActiveWindow.Document.Sentences(1).Select                 ' Επιλογή της 2ης Πρότασης Word Doc.
    ActiveDocument.ActiveWindow.Selection.Copy                     ' Αντριγραφή της επιλογής
    ActiveSheet.Range("G10").PasteSpecial xlPasteValues            ' Επικόλληση αρχίζοντας στο G10
    ActiveSheet.Range("G10").EntireColumn.ColumnWidth = 45         ' Αλλαγή Πλάτους Στήλης σε 45 στ.
    ActiveSheet.Range("G9:G10").WrapText = True                    ' Αναδίπλωση κειμένου

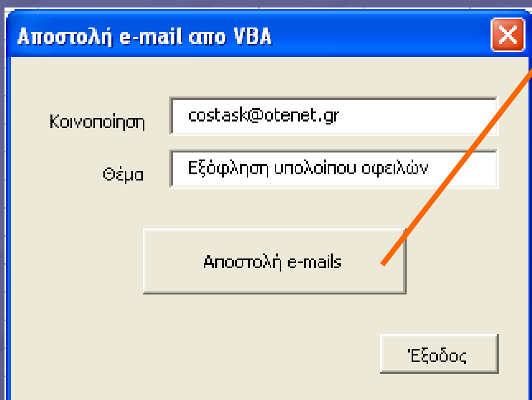
    ActiveSheet.Range("G13").Value = "Ακολουθεί η 2η ΠΡΟΤΑΣΗ Μόνο"   ' Επικεφαλίδα στο κελί G13
    ActiveSheet.Range("G13").Font.Color = RGB(0, 255, 0)            ' Πράσινο Χρώμα
    ActiveSheet.Range("G13").Font.Bold = True                       ' Bold
    ActiveSheet.Range("G13").VerticalAlignment = xlCenter           ' Κάθετη στοίχιση Κέντρο
    ActiveSheet.Range("G13").HorizontalAlignment = xlCenter         ' Οριζόντια στοίχιση Κέντρο
    ActiveSheet.Range("G14").Value = MyWd.ActiveWindow.Document.Sentences(2) 'Εναλλακτικός Τρόπος
    ActiveSheet.Range("G14").EntireColumn.ColumnWidth = 45          ' Αλλαγή Πλάτους Στήλης σε 45 στ.
    ActiveSheet.Range("G13:G14").WrapText = True                    ' Αναδίπλωση κειμένου

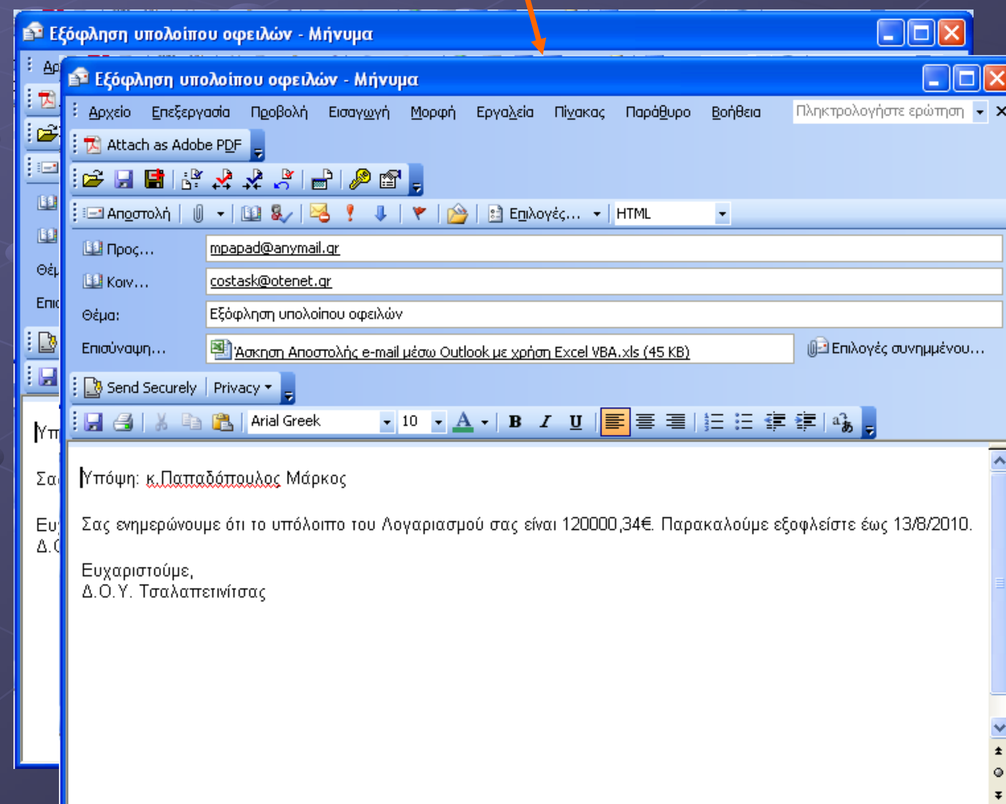
End Sub
```

Συνεργασία με άλλες Εφαρμογές

Με MS-Outlook - (Παράδειγμα) - (1/2)

	A	B	C	D	E
1	Επώνυμο	Όνομα	Διεύθυνση e-mail	Υπόλοιπο	Ημ/νία Πληρωμής
2	Παπαδόπουλος	Μάρκος	mpapad@anymail.gr	120.000,34	13/8/2010
3	Μαυρίδου	Άννα	annama@nomail.gr	12.000,00	17/9/2010
4	Πρίμος	Ιωάννης	iprimo@anymail.com	150.000,50	1/1/2011

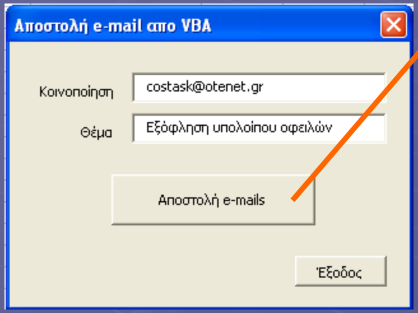




Στην άσκηση αυτή θέλουμε να γίνεται μαζική αποστολή e-mails σε όλες τις επιλεγμένες εγγραφές του Ενεργού Φύλλου, με «*Θέμα*» και «*Κοινοποίηση*» βάσει των στοιχείων που πληκτρολογεί ο χρήστης στην Φόρμα.

Συνεργασία με άλλες Εφαρμογές

Με MS-Outlook - (Παράδειγμα) - (2/2)




```
Private Sub SendMailCommandButton_Click()     ' Για το κουμπί «Αποστολή e-mails»
   Dim OutApp, OutMail As Object, i As Integer, selRow As Range

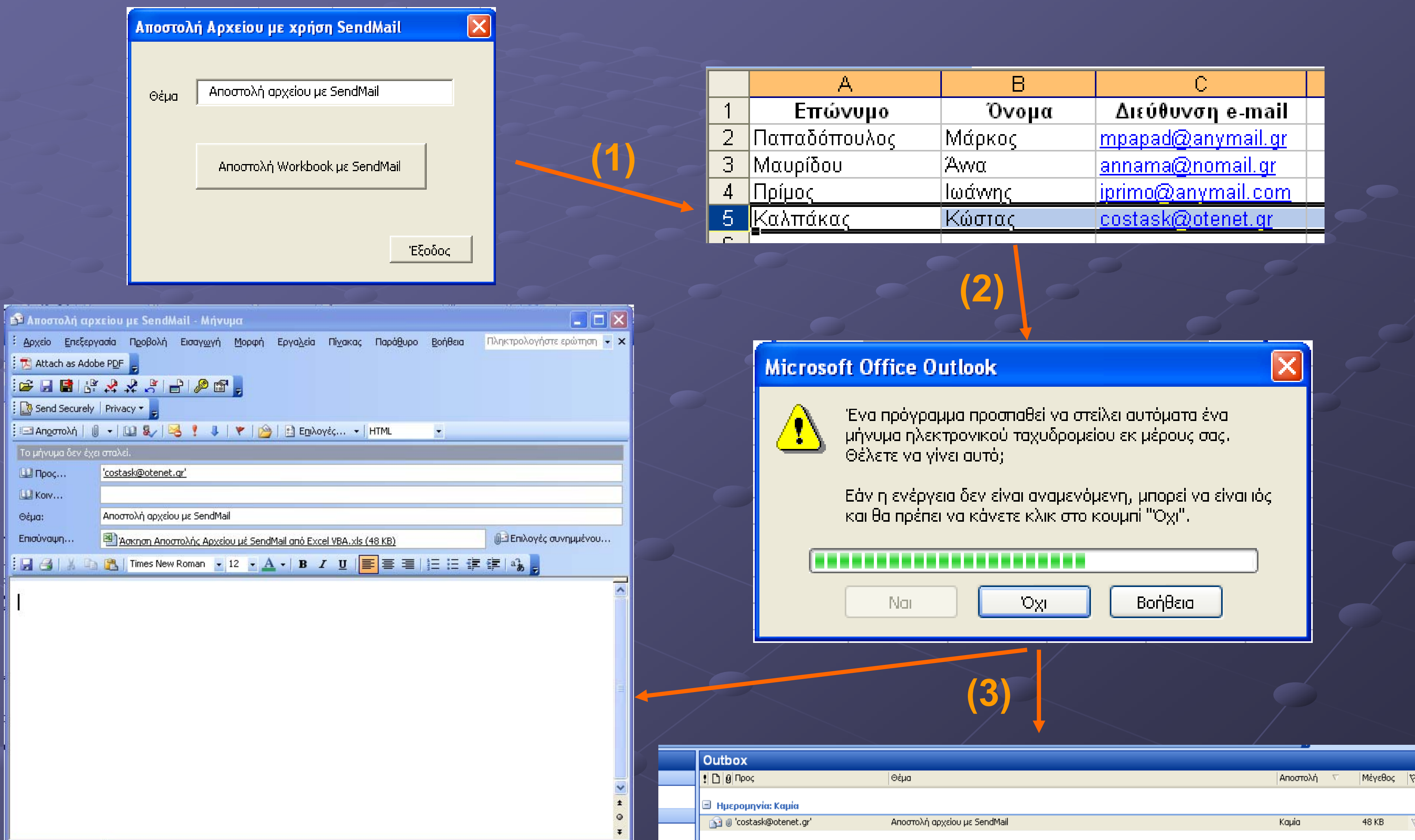
   For Each selRow In Selection.Rows                          ' Για κάθε Επιλεγμένη Γραμμή
       Set OutApp = CreateObject("Outlook.Application")
       Set OutMail = OutApp.CreateItem(0)

       On Error Resume Next
       i = selRow.Row                                         ' Το i έχει τον αριθμό Γραμμής
       If Sheets("Data").Cells(i, 1) <> "" Then               ' Για μη κενές Γραμμές
         With OutMail
           .To = Sheets("Data").Cells(i, 3).Value             ' Παραλήπτης (από την Στήλη C )
           .CC = CCTextBox.Value                              ' Κοινοποίηση
           .BCC = ""                                          ' Κρυφή Κοινοποίηση
           .Subject = SubjectTextBox.Value                    ' Θέμα
                                                     ' Το Σώμα (body) του e-mail
           .Body = "Υπόψη: κ." & Sheets("Data").Cells(i, 1).Value & " " & _
               Sheets("Data").Cells(i, 2).Value & vbCrLf & vbCrLf & _
               "Σας ενημερώνουμε ότι το υπόλοιπο του Λογαριασμού σας είναι " & _
               Sheets("Data").Cells(i,4).Value & "€.Παρακαλούμε εξοφλείστε έως " _
               & Sheets("Data").Cells(i, 5).Value & "." & vbCrLf & vbCrLf & _
               "Ευχαριστούμε," & vbCrLf & "Δ.Ο.Υ. Τσαλαπετινίτσας"
                                                              ' Επισύναψη του Active WorkBook
           .Attachments.Add ActiveWorkbook.Path & "/" & ActiveWorkbook.Name
           .Display  ' Το Μήνυμα δημιουργείται αλλά ΔΕΝ αποστέλλεται αυτόματα
           ' .Send   ' Με .Send το Μήνυμα δημιουργείται ΚΑΙ αποστέλλεται αυτόματα
       End If
       On Error GoTo 0

       Set OutMail = Nothing        ' Απελευθέρωση Αντικειμένου
       Set OutApp = Nothing         ' Απελευθέρωση Αντικειμένου
   Next selRow
End Sub
```

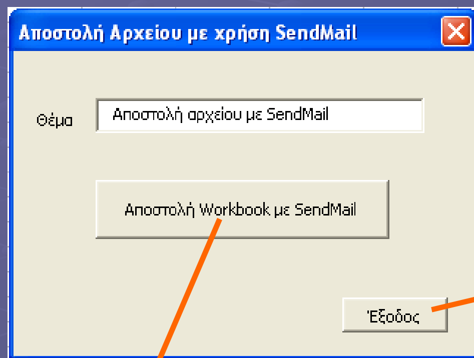
Συνεργασία με άλλες Εφαρμογές

Αποστολή αρχείου μέσω e-mail (SendMail) (1/2)



Συνεργασία με άλλες Εφαρμογές

Αποστολή αρχείου μέσω e-mail (SendMail) (2/2)




```
Private Sub ExitButton_Click()
    Unload Me
End Sub
```

```
Private Sub SendMailCommandButton_Click()
    Dim i As Integer, selRow As Range
    Dim ToStr, SubjStr As String

    For Each selRow In Selection.Rows                   ' Για κάθε Επιλεγμένη Γραμμή
        On Error Resume Next
        i = selRow.Row                                  ' Το i έχει τον αριθμό Γραμμής
        If Sheets("Data").Cells(i, 1) <> "" Then        ' Για μη κενές Γραμμές
            ToStr = Sheets("Data").Cells(i, 3).Value    ' Παραλήπτης
            SubjStr = SubjectTextBox.Value              ' Θέμα
            ActiveWorkbook.SendMail ToStr, SubjStr, False
        End If
    Next selRow
End Sub
```

Μελέτη Περίπτωσης

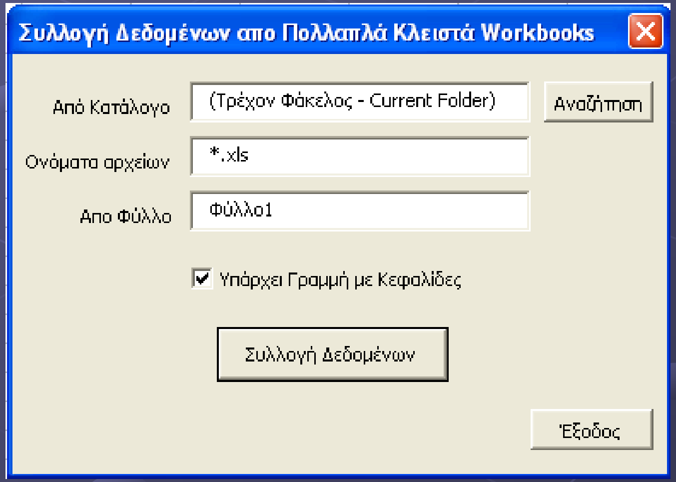
Στο πλαίσιο της μελέτης περίπτωσης θα αναπτύξουμε μια μικρή Εφαρμογή σε VBA για την συλλογή δεδομένων από πολλά Βιβλία Εργασίας Excel και την συγκέντρωσή τους στο Ενεργό Φύλλο.

Αυτό είναι κάτι το οποίο αφ' ενός μεν μπορεί να φανεί ιδιαίτερα χρήσιμο στην καθημερινή εργασία οποιουδήποτε εργαζομένου στο Δημόσιο Τομέα αφετέρου δε θα μας δώσει την ευκαιρία να επαναλάβουμε και να ολοκληρώσουμε πολλά από τα βασικά σημαία της VBA, όπως είναι:

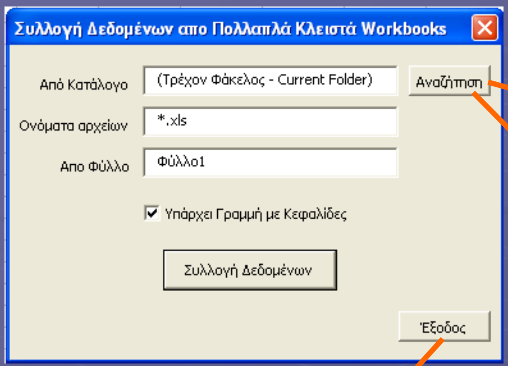
- Η Δημιουργία και παραμετροποίηση Φορμών και Στοιχείων Ελέγχου
- Η χρήση του Μοντέλου Αντικειμένων του Excel
- Οι Μεταβλητές του Excel και ο τρόπος δήλωσης και χρήσης τους
- Η χρήση των Αλγοριθμικών Δομών της VBA
- Η Δημιουργία και Κλήση Συναρτήσεων
- Η Συλλογή δεδομένων από κλειστά Βιβλία του Excel
- Η χρήση στοιχείων του Λειτουργικού Συστήματος (όπως τα Παράθυρα Διαλόγου των Windows και η πρόσβαση στο Σύστημα Αρχείων – File System)

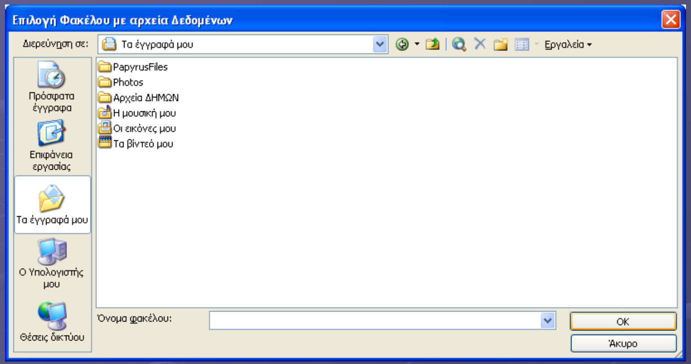
Μελέτη Περίπτωσης
Επαναληπτική Άσκηση (1/7)

Στην παρακάτω Φόρμα, θέλουμε πατώντας το κουμπί «*Συλλογή Δεδομένων*» να αντιγράφονται αυτόματα στο Ενεργό Φύλλο (ActiveSheet), όλες οι μη κενές Γραμμές των αρχείων δεδομένων (Excel) που βρίσκονται στον επιλεγμένο Φάκελο και ικανοποιούν τα δοθέντα κριτήρια για το όνομα αρχείου. Τα δεδομένα θα λαμβάνονται από εκείνο το Φύλλο που θα καταχωρήσει ο χρήστης στην Φόρμα. Εάν το CheckBox «*Υπάρχει Γραμμή με Κεφαλίδες*» είναι επιλεγμένο, τότε δεν θα αντιγράφεται η πρώτη Γραμμή από κάθε αρχείο (αφού θα περιέχει Κεφαλίδες και όχι δεδομένα). Πατώντας στο κουμπί «Αναζήτηση» θα εμφανίζεται ο Διάλογος Επιλογής Φακέλου των Windows, από τον οποίο ο Χρήστης θα μπορεί να επιλέξει τον Φάκελο που περιέχει τα αρχεία με τα δεδομένα. Είναι προφανές ότι για να υπάρχει πρακτική εφαρμογή, θα πρέπει όλα τα αρχεία δεδομένων να έχουν την ίδια γραμμογράφηση (τα ίδια ακριβώς Πεδία με την ίδια σειρά).



Μελέτη Περίπτωσης
Επαναληπτική Άσκηση (2/7)






```
Private Sub BrowseForFolderCommandButton_Click()
    DirectoryTextBox.Value = GetFolder(CurDir)
End Sub
```

```
Private Sub ExitButton_Click()
    Unload Me
End Sub
```

```
' Συνάρτηση που εμφανίζει το Παράθυρο επιλογής Φακέλου των Windows και επιστρέφει
' τον Κατάλογο που θα επιλέξει ο χρήστης.
Private Function GetFolder(InitialPath As String) As String
    Dim fldr As FileDialog
    Dim sItem As String

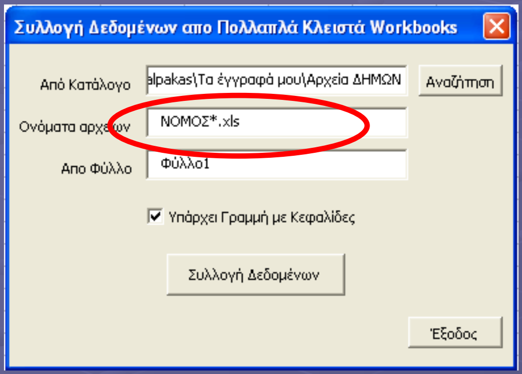
    Set fldr = Application.FileDialog(msoFileDialogFolderPicker)

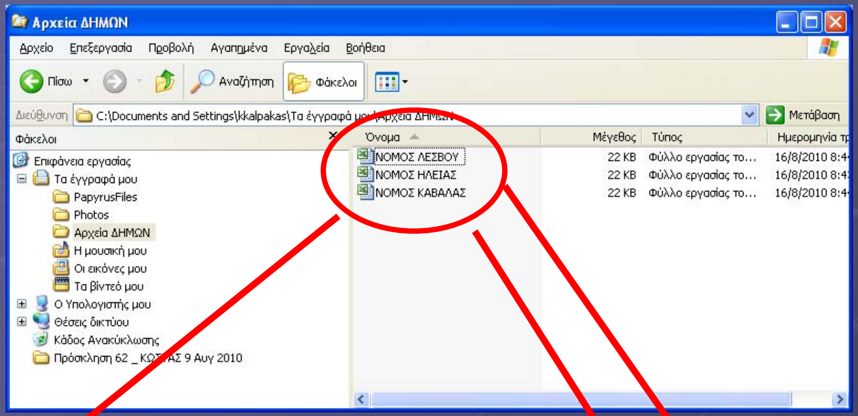
    With fldr
        .Title = "Επιλογή Φακέλου με Αρχεία Δεδομένων"   ' Τίτλος Παραθύρου
        .AllowMultiSelect = False                      ' Επιλογή μόνο ενός Φακέλου
        .InitialFileName = InitialPath                 ' Αρχικός Φάκελος
        If .Show <> -1 Then GoTo NextCode
        sItem = .SelectedItems(1)                      ' Επιλεγείς Φάκελος
    End With

NextCode:
    GetFolder = sItem                   ' Επιστροφή του επιλεγέντος Φακέλου
    Set fldr = Nothing                  ' Απελευθέρωση του αντικειμένου
End Function
```

Πατώντας το κουμπί «*Αναζήτηση*» εμφανίζεται το Παράθυρο Διαλόγου των Windows για επιλογή αρχείων

Μελέτη Περίπτωσης
Επαναληπτική Άσκηση (3/7)





Τα δεδομένα βρίσκονται σε 3 διαφορετικά και «κλειστά» Βιβλία Εργασίας.

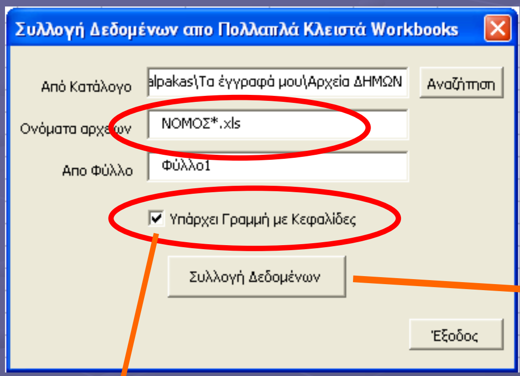
	A	B	C	D	E	F	G
1	Νομός	Δήμος	Πληθυσμός	Αριθμός Μαθητών	Αριθμός Δημοτικών	Αριθμός Γυμνασίων	Αριθμός Λυκείων
2	ΛΕΣΒΟΥ	Μυτιλήνης	36.196	3.600	15	9	5
3	ΛΕΣΒΟΥ	Πλωμαρίου	6.698	710	2	1	1
4	ΛΕΣΒΟΥ	Καλλονής	8.194	780	2	1	1
5							

	A	B	C	D	E	F	G
1	Νομός	Δήμος	Πληθυσμός	Αριθμός Μαθητών	Αριθμός Δημοτικών	Αριθμός Γυμνασίων	Αριθμός Λυκείων
2	ΚΑΒΑΛΑΣ	Καβάλλας	63.293	5.800	20	15	10
3	ΚΑΒΑΛΑΣ	Ελευθερούπολης	11.401	950	3	2	1
4	ΚΑΒΑΛΑΣ	Θάσου	13.765	1.450	5	3	2
5	ΚΑΒΑΛΑΣ	Παγγαίου	4.764	345	1	1	1
6							

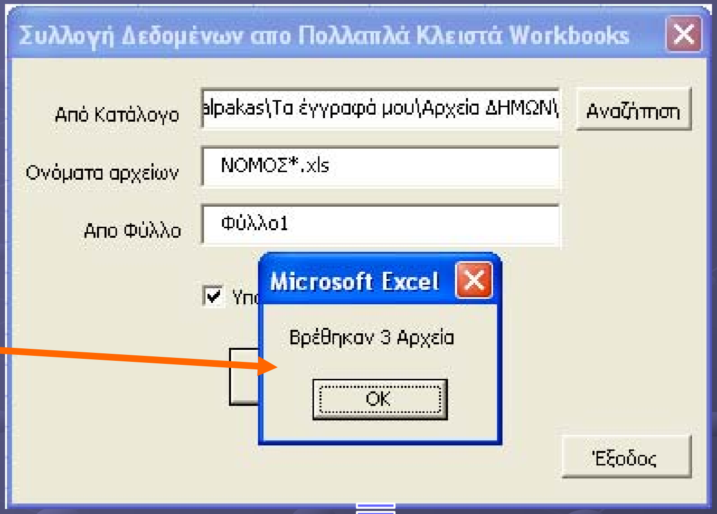
	A	B	C	D	E	F	G
1	Νομός	Δήμος	Πληθυσμός	Αριθμός Μαθητών	Αριθμός Δημοτικών	Αριθμός Γυμνασίων	Αριθμός Λυκείων
2	ΗΛΕΙΑΣ	Πύργου	35.681	3.400	10	8	8
3	ΗΛΕΙΑΣ	Γαστούνης	11.460	1.200	3	2	1
4	ΗΛΕΙΑΣ	Ζαχάρως	11.841	1.080	3	2	1
5	ΗΛΕΙΑΣ	Ωλένης	9.046	830	2	1	1
6							

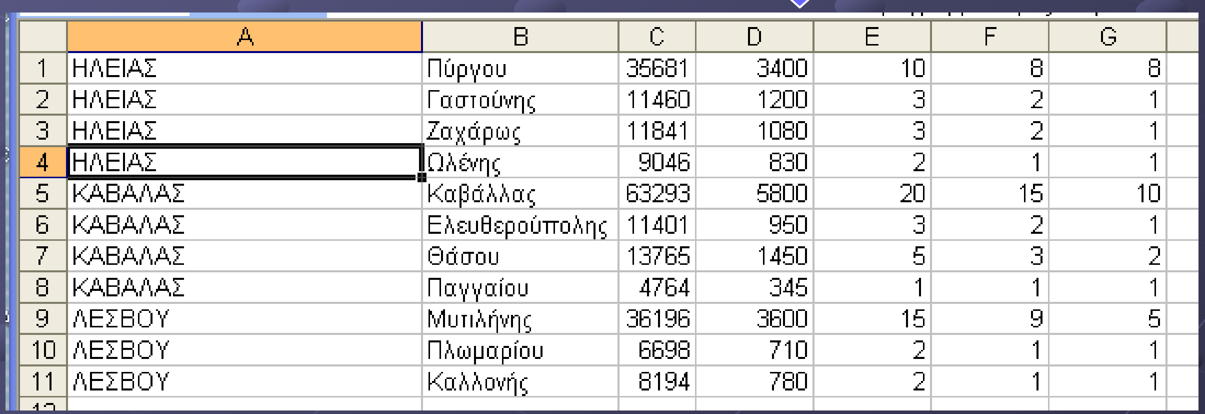
Κ. Καλπάκας - costask@otenet.gr

Μελέτη Περίπτωσης
Επαναληπτική Άσκηση (4/7)



Οι Επικεφαλίδες **δεν** θα αντιγραφούν

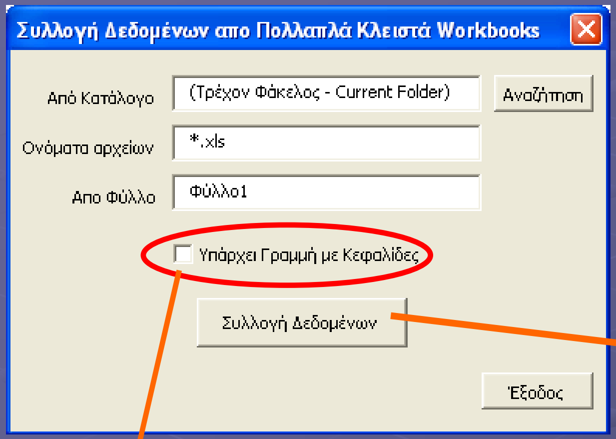


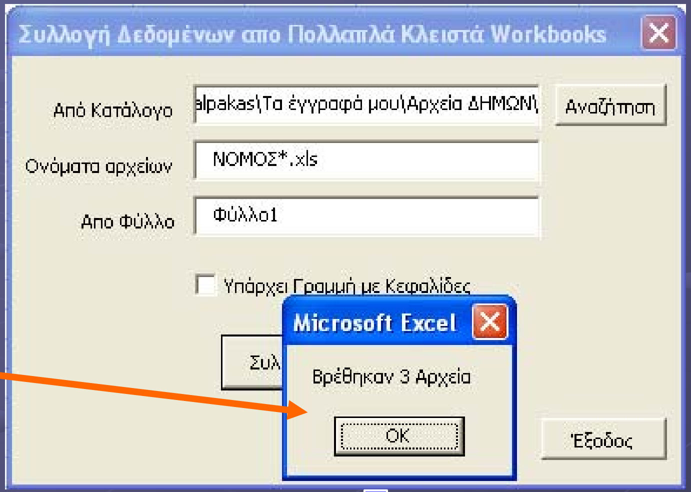


	A	B	C	D	E	F	G
1	ΗΛΕΙΑΣ	Πύργου	35681	3400	10	8	8
2	ΗΛΕΙΑΣ	Γαστούνης	11460	1200	3	2	1
3	ΗΛΕΙΑΣ	Ζαχάρως	11841	1080	3	2	1
4	ΗΛΕΙΑΣ	Ωλένης	9046	830	2	1	1
5	ΚΑΒΑΛΑΣ	Καβάλλας	63293	5800	20	15	10
6	ΚΑΒΑΛΑΣ	Ελευθερούπολης	11401	950	3	2	1
7	ΚΑΒΑΛΑΣ	Θάσου	13765	1450	5	3	2
8	ΚΑΒΑΛΑΣ	Παγγαίου	4764	345	1	1	1
9	ΛΕΣΒΟΥ	Μυτιλήνης	36196	3600	15	9	5
10	ΛΕΣΒΟΥ	Πλωμαρίου	6698	710	2	1	1
11	ΛΕΣΒΟΥ	Καλλονής	8194	780	2	1	1

Τα περιεχόμενα των τριών Αρχείων που ταιριάζουν στο «ΝΟΜΟΣ*.xls» αντιγράφηκαν στο ActiveSheet

Μελέτη Περίπτωσης
Επαναληπτική Άσκηση (5/7)





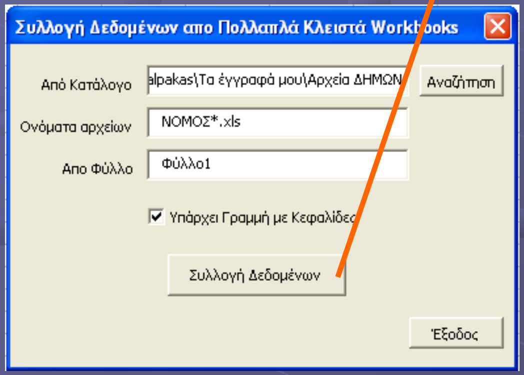
Οι Επικεφαλίδες **θα** αντιγραφούν

	A	B	C	D	E	F	G	H
1	Νομός	Δήμος	Πληθυ	Αριθμός Μ	Αριθμός Δ	Αριθμός Γ	Αριθμός Α	
2	ΗΛΕΙΑΣ	Πύργου	35681	3400	10	8	8	
3	ΗΛΕΙΑΣ	Γαστούνης	11460	1200	3	2	1	
4	ΗΛΕΙΑΣ	Ζαχάρως	11841	1080	3	2	1	
5	ΗΛΕΙΑΣ	Ωλένης	9046	830	2	1	1	
6	Νομός	Δήμος	Πληθυ	Αριθμός Μ	Αριθμός Δ	Αριθμός Γ	Αριθμός Α	
7	ΚΑΒΑΛΑΣ	Καβάλλας	63293	5800	20	15	10	
8	ΚΑΒΑΛΑΣ	Ελευθερούπολης	11401	950	3	2	1	
9	ΚΑΒΑΛΑΣ	Θάσου	13765	1450	5	3	2	
10	ΚΑΒΑΛΑΣ	Παγγαίου	4764	345	1	1	1	
11	Νομός	Δήμος	Πληθυ	Αριθμός Μ	Αριθμός Δ	Αριθμός Γ	Αριθμός Α	
12	ΛΕΣΒΟΥ	Μυτιλήνης	36196	3600	15	9	5	
13	ΛΕΣΒΟΥ	Πλωμαρίου	6698	710	2	1	1	
14	ΛΕΣΒΟΥ	Καλλονής	8194	780	2	1	1	
15								

Τα περιεχόμενα των τριών Αρχείων που ταιριάζουν στο «ΝΟΜΟΣ*.xls» αντιγράφηκαν στο ActiveSheet

Μελέτη Περίπτωσης

Επαναληπτική Άσκηση (6/7)




```
Private Sub CollectDataButton_Click()            ' Κουμπί «Συλλογή Δεδομένων»
    Dim SrcDir, SrcFile, FromSheet, FromRow, DataSource, ToRow, FileFilter As String
    Dim SelectedFiles As Variant
    Dim i, j As Long

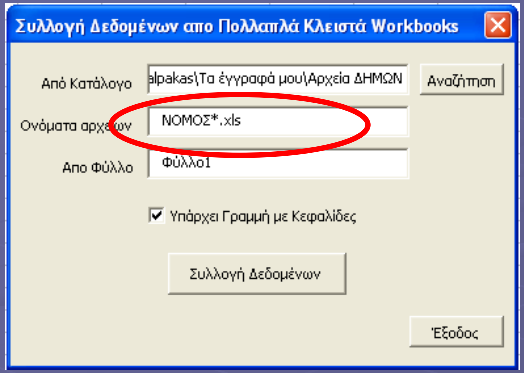
    ActiveSheet.UsedRange.Clear                   ' Καθαρισμός Φύλλου προορισμού
    SrcDir = DirectoryTextBox.Value               ' Φάκελος με αρχεία δεδομένων
    If SrcDir = "(Τρέχον Φάκελος - Current Folder)" Then  ' Εάν επιλεγεί ο Τρέχον Φάκελος
        SrcDir = CurDir & "\"
    ElseIf (Right(SrcDir, 1) <> "\") Then         ' Έλεγχος για «\» στο τέλος και προσθήκη.
        SrcDir = SrcDir & "\"
    End If
    DirectoryTextBox.Value = SrcDir          ' Εμφάνιση του Path του επιλεγέντος Φακέλου.

    SrcFile = FileNameTextBox.Value          ' Ονόματα αρχείων δεδομένων
    FromSheet = FromSheetTextBox.Value       ' Όνομα Φύλλου για (όλα) τα αρχείων δεδομένων
    FileFilter = SrcDir & SrcFile            ' Φίλτρο Ονόματος για τα αρχεία δεδομένων
    SelectedFiles = GetFileList(FileFilter)  ' Get the files that match the criteria

    If IsArray(SelectedFiles) Then           ' Εφόσον βρεθούν αρχεία με αυτό το όνομα
        MsgBox "Βρέθηκαν " & UBound(SelectedFiles) & " Αρχεία"
        LastLine = 1          ' Μετρητής Συνολικών γραμμών που προστίθενται στο ActiveSheet
        For i = LBound(SelectedFiles) To UBound(SelectedFiles)  ' Για κάθε αρχείο
            If HasHeadersCheckBox.Value = True Then             ' Αν υπάρχουν Κεφαλίδες
                j = 1                        ' Μετρητής Γραμμών αρχείου δεδομένων
            Else
                j = 0
            End If
            HasMoreData = True

            While HasMoreData = True                            ' Όσο υπάρχουν δεδομένα
                FromRow = (1 + j) & ":" & (1 + j)               ' Γραμμή προς αντιγραφή
                DataSource = "= '" & SrcDir & "[" & SelectedFiles(i) & "]" & _
                             FromSheet & "'!" & FromRow
              MsgBox ("DataSource:" & DataSource)
                ToRow = LastLine & ":" & LastLine               ' Γραμμή προς επικόλληση
                ActiveSheet.Range(ToRow) = DataSource           ' Αντιγραφή στο ActiveSheet
                If ActiveSheet.Cells(LastLine, 1) = 0 Then
                    HasMoreData = False                        ' Δεν υπάρχουν άλλα δεδομένα
                                  ' Διαγραφή τελευταίας Γραμμής που περιέχει μόνο μηδενικά
                    ActiveSheet.Rows(LastLine).EntireRow.Delete
                Else
                    LastLine = LastLine + 1      ' Αύξηση μετρητή Συνολικών Γραμμών
                    j = j + 1                    ' Αύξηση μετρητή Γραμμών αρχείου Δεδομένων
                End If
            Wend
        Next i
    Else                   ' Δεν βρέθηκαν αρχεία που να ταιριάζουν στο φίλτρο Ονόματος
        MsgBox "No matching Files where found!"
    End If
End Sub
```

Μελέτη Περίπτωσης
Επαναληπτική Άσκηση (7/7)



Η παρακάτω Συνάρτηση επιστρέφει έναν Πίνακα με τα ονόματα των Αρχείων που ικανοποιούν το Φίλτρο που έδωσε ο χρήστης στην Φόρμα. Αν δεν υπάρχει κανένα τέτοιο Αρχείο επιστρέφει «False»

```
' Συνάρτηση που επιστρέφει ένα Πίνακα (Array) με ονόματα Αρχείων που ικανοποιούν
' το δοθέν φίλτρο ή «False» εάν δεν βρεθεί κανένα τέτοιο αρχείο.
Private Function GetFileList(Filter As String) As Variant
    Dim FileArray() As Variant
    Dim FileCount As Integer
    Dim FileName As String

    FileCount = 0                              ' Μετρητής αρχείων που ταιριάζουν
    FileName = Dir(Filter)
    If FileName = "" Then                      ' Εάν δεν υπάρχουν αρχεία που ταιριάζουν
        GetFileList = False                    ' Θα επιστρέψει False
        Exit Function
    Else
        Do While FileName <> ""                ' Επανάλαβε όσο υπάρχουν αρχεία
            FileCount = FileCount + 1                     ' Αύξηση μετρητή αρχείων
            ReDim Preserve FileArray(1 To FileCount)      ' Αύξηση μεγέθους Array
            FileArray(FileCount) = FileName               ' Προσθήκη Ονόματος στο Array
            FileName = Dir()                   ' Λήψη επόμενου Αρχείου που ταιριάζει
        Loop
        GetFileList = FileArray                ' Επιστροφή Array με τα Ονόματα Αρχείων
    End If
End Function
```

Ερωτήσεις
?

Καλή Συνέχεια

Κώστας Καλπάκας
e-mail: costask@otenet.gr
Κιν.: 6944 261906